Development of Methods for Identifying Key Variables in Complex Mathematical Models of Biological Systems

Thesis submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy (PhD)

Author: Ruby Khan

Supervisor: Dr hab. inż. Krzysztof Puszyński, prof. Pol. Śl.
(PhD, DSc, Professor)

Department/Faculty: Department of Systems Biology and Engineering

Year of Doctoral Training/Semester: 4th year

Email Address: ruby.khan@polsl.pl

Discipline: Biomedical Engineering

Keywords: Systems Biology, Ovarian Cancer, Boolean Networks, PageRank, Random Walks, RCNNs, NF-κB, ATM, Network Dynamics, Data Integration, Modeling.

Date of Submission: 2nd July 2025

Silesian University of Technology, Gliwice, Poland
Department of Systems Biology and Engineering

Abstract

Background: One of the central objectives of systems biology is the integration of high-throughput omics data with computational modeling to better understand molecular interactions and cellular mechanisms. The intricate nature of genomic, transcriptomic, and proteomic networks presents substantial challenges, often limiting the ability of traditional models to capture the dynamic behavior of biological systems. Therefore, advanced analytical approaches are necessary to elucidate these complex interactions and identify key regulatory elements within biological networks.

Objective: This study aims to establish a comprehensive computational pipeline for automated data extraction, curation, and analysis, facilitating the identification of crucial variables within complex biological networks. By leveraging multi-omics data from repositories such as Pathway Commons, AnimalTFDB, and the Genomics Data Commons (GDC), the research seeks to elucidate transcription factor (TF) to ligand-receptor (L-R), protein-protein (P-P), and gene-gene (G-G) interactions. The Cancer Genome Atlas (TCGA) ovarian cancer dataset serves as a case study to validate the approach, initially focusing on the connection between the NF-κB and **p53** pathways, followed by additional network analyses of the **Cell Cycle** and **MAPK Signaling** pathways to identify key regulatory nodes.

Methodology: The research was conducted through a series of structured stages:

1. **Data Extraction:** A Python-based pipeline was employed to extract large-scale biological data from databases including Pathway Commons, AnimalTFDB, and CellTalkDB. Python libraries such as `pandas` and `requests` were utilized for data manipulation and automated API access. The data were filtered based on relevant identifiers, such as KEGG and PubMed IDs, to ensure high-quality interactions for further analysis.

2. **Database Design:** The curated datasets were systematically organized in a custom-designed MySQL relational database to facilitate efficient data management and retrieval. The database schema captured various types of biological interactions (e.g., TF-to-L-R, P-P, G-G) with foreign key relationships ensuring data integrity.

3. **Data Filtering and Network Construction:** The filtered data were used to construct a directed network graph, representing molecular interactions among genes, proteins, and signaling pathways. Edge weights were assigned probabilistically to reflect the significance of each interaction.

4. **Network Visualization and Initial Analysis in Cytoscape:** The network's initial visualization was performed using Cytoscape, allowing the exploration of structural properties such as clustering coefficients, average path lengths, and degree distributions. This analysis provided insights into the network's topological features, identifying regions of high connectivity and key nodes.

5. **Advanced Computational Analysis in Python:** The network was subsequently analyzed using Python-based techniques to uncover critical regulatory elements:

- **Boolean Network Modeling** simulated regulatory dynamics to identify pivotal nodes and stable states critical to cellular decision-making processes.
- **PageRank Algorithm** assessed node centrality to highlight influential regulatory elements within the network.
- **Random Walk Modeling** identified key nodes by simulating stochastic flow across the network.
- **Recurrent Convolutional Neural Networks (RCNNs)** captured temporal dependencies to predict critical regulatory elements from time-series data, enhancing dynamic signaling insights.

To validate the framework, the same pipeline was applied to the **Cell Cycle** and **MAPK Signaling** pathways. These pathways were chosen due to their critical roles in cancer progression and cellular regulation. The analysis included random walk simulations, Boolean modeling, PageRank analysis, centrality measures, and motif analysis to identify key nodes and their functional roles.

Results: The multi-stage analysis provided comprehensive insights into the ovarian cancer network. Key regulatory nodes such as *NF-κB*, *p53*, *ATM*, and *TNFR1* emerged as central to network stability. Specific findings include:

- **Boolean Network Modeling** revealed crucial nodes like *NF-κB*, *p53*, *IKKα*, and *ATM*, alongside significant mRNA components (*A20 mRNA*, *Wip1 mRNA*, *PTEN mRNA*).
- **PageRank Analysis** identified prominent nodes including *NF-κB*, phosphorylated *ATM (ATM-p)*, and phosphorylated *Chk2 (Chk2-p)*.
- **Random Walk Modeling** highlighted nodes such as *p53*, *ATM*, and *NF-κB*, indicating their significance in processes like DNA damage response and apoptosis.
- **RCNN Analysis** emphasized regulatory elements such as *TNFR1*, *IKKα*, and apoptosis-associated proteins (*Bax*, *p21*).

The analysis of the **Cell Cycle** and **MAPK Signaling** pathways further validated the framework. Key nodes identified in the **Cell Cycle** pathway included *Cyclin D*, *ATM/ATR*, *p53*, and *CDK4/6*, which are critical for cell cycle regulation and DNA damage response. In the **MAPK Signaling** pathway, nodes such as *EGFR*, *RAS*, *MEK*, and *ERK* were identified as central to cellular proliferation and differentiation. These findings highlight the potential of these nodes as therapeutic targets in cancer treatment.

Conclusion: This research advances the field by presenting a comprehensive, integrative computational pipeline for network analysis and modeling, enabling precise identification of regulatory elements. The framework provides novel insights into gene regulation, signaling pathways, and potential therapeutic targets, with broader implications for personalized medicine and disease modeling. The successful application of the pipeline to the **Cell Cycle** and **MAPK Signaling** pathways underscores its versatility and reliability in identifying critical nodes across diverse biological networks. These findings suggest that the identified nodes, such as *ATM/ATR*, *p53*, *EGFR*, and *ERK*, should be prioritized for further investigation in drug discovery efforts aimed at targeting cancer-related pathways.

Keywords: biomedical engineering, systems biology, network modeling, Boolean network, PageRank, random walk, RCNN, gene regulation, therapeutic targets, personalized medicine.

Streszczenie

Tło: Jednym z głównych celów biologii systemów jest integracja danych omicznych wysokiej przepustowości z modelowaniem obliczeniowym, aby lepiej zrozumieć interakcje molekularne i mechanizmy komórkowe. Złożona natura sieci genomowych, transkryptomicznych i proteomicznych stawia istotne wyzwania, często ograniczając zdolność tradycyjnych modeli do uchwycenia dynamicznego zachowania systemów biologicznych. Dlatego zaawansowane podejścia analityczne są niezbędne do wyjaśnienia tych złożonych interakcji i identyfikacji kluczowych elementów regulacyjnych w sieciach biologicznych.

Cel: Niniejsze badanie ma na celu opracowanie kompleksowego potoku obliczeniowego do automatycznego pozyskiwania, porządkowania i analizy danych, ułatwiającego identyfikację kluczowych zmiennych w złożonych sieciach biologicznych. Wykorzystując dane multi-omics z repozytoriów takich jak Pathway Commons, AnimalTFDB i Genomics Data Commons (GDC), badanie ma na celu wyjaśnienie interakcji czynników transkrypcyjnych (TF) z ligandami-receptorami (L-R), białko-białko (P-P) i gen-gen (G-G). Zestaw danych raka jajnika z The Cancer Genome Atlas (TCGA) służy jako studium przypadku do walidacji podejścia, początkowo koncentrując się na połączeniu między szlakami NF**-**κB i **p53**, a następnie przeprowadzając dodatkowe analizy sieciowe szlaków **Cyklu Komórkowego** i **Sygnałowego MAPK** w celu identyfikacji kluczowych węzłów regulacyjnych.

Metodologia: Badanie przeprowadzono w serii ustrukturyzowanych etapów:

1. **Pozyskiwanie danych:** Zastosowano potok oparty na Pythonie do ekstrakcji danych biologicznych na dużą skalę z baz danych, w tym Pathway Commons, AnimalTFDB i CellTalkDB. Biblioteki Pythona, takie jak `pandas` i `requests`, zostały wykorzystane do manipulacji danymi i automatycznego dostępu do API. Dane zostały przefiltrowane na podstawie odpowiednich identyfikatorów, takich jak KEGG i PubMed ID, aby zapewnić wysoką jakość interakcji do dalszej analizy.

2. **Projekt bazy danych:** Zgromadzone zestawy danych zostały uporządkowane w zaprojektowanej relacyjnej bazie danych MySQL, aby ułatwić efektywne zarządzanie danymi i ich pobieranie. Schemat bazy danych przechwytywał różne typy interakcji biologicznych (np. TF-do-L-R, P-P, G-G) z relacjami klucza obcego zapewniającymi integralność danych.

3. **Filtrowanie danych i konstrukcja sieci:** Przefiltrowane dane zostały wykorzystane do skonstruowania skierowanego grafu sieciowego, reprezentującego interakcje molekularne między genami, białkami i szlakami sygnałowymi. Wagi krawędzi zostały przypisane probabilistycznie, aby odzwierciedlić znaczenie każdej interakcji.

4. **Wizualizacja sieci i wstępna analiza w Cytoscape:** Wstępną wizualizację sieci wykonano przy użyciu Cytoscape, umożliwiając eksplorację właściwości strukturalnych, takich jak współczynniki klastrowania, średnie długości ścieżek i rozkłady stopni. Ta analiza dostarczyła informacji o cechach topologicznych sieci, identyfikując regiony o wysokiej łączności i kluczowe węzły.

5. **Zaawansowana analiza obliczeniowa w Pythonie:** Sieć została następnie przeanalizowana przy użyciu technik opartych na Pythonie w celu odkrycia krytycznych elementów regulacyjnych:

- **Modelowanie sieci boolowskich** symulowało dynamikę regulacyjną w celu identyfikacji kluczowych węzłów i stanów stabilnych istotnych dla procesów decyzyjnych komórki.
- **Algorytm PageRank** oceniał centralność węzłów, aby wyróżnić wpływowe elementy regulacyjne w sieci.
- **Modelowanie losowych spacerów** identyfikowało kluczowe węzły poprzez symulację stochastycznego przepływu przez sieć.
- **Rekurencyjne konwolucyjne sieci neuronowe (RCNNs)** wychwytywały zależności czasowe, aby przewidywać krytyczne elementy regulacyjne z danych szeregów czasowych, zwiększając zrozumienie dynamicznej sygnalizacji.

Aby zweryfikować framework, ten sam potok został zastosowany do szlaków **Cyklu Komórkowego** i **Sygnałowego MAPK**. Te szlaki zostały wybrane ze względu na ich kluczową rolę w progresji nowotworu i regulacji komórkowej. Analiza obejmowała symulacje losowych spacerów, modelowanie boolowskie, analizę PageRank, miary centralności i analizę motywów w celu identyfikacji kluczowych węzłów i ich ról funkcjonalnych.

Wyniki: Wieloetapowa analiza dostarczyła kompleksowych informacji o sieci raka jajnika. Kluczowe węzły regulacyjne, takie jak *NF-κB*, *p53*, *ATM* i *TNFR1*, okazały się centralne dla stabilności sieci. Konkretne wyniki obejmują:

- **Modelowanie sieci boolowskich** ujawniło kluczowe węzły, takie jak *NF-κB*, *p53*, *IKKα* i *ATM*, wraz z istotnymi składnikami mRNA (*A20 mRNA*, *Wip1 mRNA*, *PTEN mRNA*).
- **Analiza PageRank** zidentyfikowała prominentne węzły, w tym *NF-κB*, fosforylowane *ATM (ATM-p)* i fosforylowane *Chk2 (Chk2-p)*.
- **Modelowanie losowych spacerów** podkreśliło węzły takie jak *p53*, *ATM* i *NF-κB*, wskazując na ich znaczenie w procesach takich jak odpowiedź na uszkodzenie DNA i apoptoza.
- **Analiza RCNN** uwydatniła elementy regulacyjne takie jak *TNFR1*, *IKKα* i białka związane z apoptozą (*Bax*, *p21*).

Analiza szlaków **Cyklu Komórkowego** i **Sygnałowego MAPK** dodatkowo potwierdziła słuszność frameworka. Kluczowe węzły zidentyfikowane w szlaku **Cyklu Komórkowego** obejmowały *Cyklina D*, *ATM/ATR*, *p53* i *CDK4/6*, które są kluczowe dla regulacji cyklu komórkowego i odpowiedzi na uszkodzenie DNA. W szlaku **Sygnałowym MAPK**, węzły takie jak *EGFR*, *RAS*, *MEK* i *ERK* zostały zidentyfikowane jako centralne dla proliferacji i różnicowania komórek. Te wyniki podkreślają potencjał tych węzłów jako celów terapeutycznych w leczeniu nowotworów.

Wnioski: To badanie posuwa dziedzinę do przodu, przedstawiając kompleksowy, integracyjny potok obliczeniowy do analizy i modelowania sieci, umożliwiający precyzyjną identyfikację elementów regulacyjnych. Framework dostarcza nowych informacji o regulacji genów, szlakach sygnałowych i potencjalnych celach terapeutycznych, z szerszymi implikacjami dla medycyny personalizowanej i modelowania chorób. Pomyślne zastosowanie potoku do szlaków **Cyklu Komórkowego** i **Sygnałowego MAPK** podkreśla jego wszechstronność i niezawodność w identyfikacji krytycznych węzłów w różnych sieciach biologicznych. Te wyniki sugerują, że zidentyfikowane węzły, takie jak *ATM/ATR*, *p53*, *EGFR* i *ERK*, powinny być priorytetowe w dalszych badaniach nad odkrywaniem leków ukierunkowanych na szlaki związane z nowotworami.

Słowa kluczowe: inżynieria biomedyczna, biologia systemów, modelowanie sieci, sieć boolowska, PageRank, losowy spacer, RCNN, regulacja genów, cele terapeutyczne, medycyna personalizowana.

Acknowledgments

I would like to express my sincere gratitude to Allah Almighty for providing me with the guidance and strength necessary to complete this research.

My deepest thanks go to my supervisor, Professor Krzysztof Puszyński, for his exceptional scientific insight, detailed reviews, and constructive feedback. His expertise and timely input have been crucial in refining and finalizing this work. Additionally, his support in laboratory preparation, guidance on student management, and assistance with reviewing student reports have been invaluable. I deeply appreciate his comprehensive guidance throughout this process.

I am also grateful to the Polish Ministry and its leadership for their generous financial support, which was essential for the successful completion of my studies in Poland.

I wish to acknowledge the Silesian University's steadfast support and responsiveness to my needs during my time here.

Special thanks to Ms. Aleksandra Szczerbik, the department secretary, for her invaluable assistance with official documents and her support in navigating the Polish bureaucracy. Her help was instrumental in the successful completion and submission of my work.

I also extend my gratitude to Professor Jarosław Śmieja for his guidance in laboratory preparation and his instruction in the bachelor's student classes, which provided me with essential knowledge and skills, bridging the gap in my background in medical science.

I am grateful to my colleagues and the PhD and Postdoc candidates in my department for their support and encouragement, which greatly enhanced my academic experience.

Finally, but most importantly, my sincere appreciation goes to my family for their unwavering support and understanding throughout my studies. As the only female from my area to pursue a PhD abroad, my achievements have been greatly supported by my parents, Mr. and Mrs. Khan Bahadar, whose encouragement and sacrifices have been fundamental to my success. My gratitude also extends to my siblings for their steadfast support.

Thank you all for your significant contributions to this journey.

Contents

List of Publications and Communications

Publications Related to PhD

1. Khan, R., Pari, B., & Puszynski, K. (2024). Comprehensive Bioinformatic Investigation of TP53 Dysregulation in Diverse Cancer Landscapes. *Genes*, 15(5), 577. **Published**.
2. Khan, R., Khan, S., Pari, B., & Puszynski, K. (2025). Optimizing Machine Learning for Network Inference through Comparative Analysis of Model Performance in Synthetic and Real-World Networks. **Accepted in *Scientific Reports*, Springer.**
3. Khan, R., Khan, S., Pari, B., Almohaimeed, H. M., & Puszynski, K. (2025). Vancomycin-Resistant and Multidrug-Resistant Bacteria in Raw Milk Pose Critical Public Health Risks. **Under review in *Scientific Reports*, Springer.**

For more information and access to my other publications, visit my Google Scholar Profile.

International Conferences Attended

1. XXVI Gliwice Scientific Meetings, 2021
   - Organized by: Silesian University of Technology, Center of Oncology in Gliwice, and Association for the Support of Cancer Research (SWBR)
   - Role: Poster Presentation
2. XXVII Gliwice Scientific Meetings, 2023
   - Event: 70th Anniversary of DNA Helix: Nucleic Acid Structure Meets Function (Nov 16-17, 2023)
   - Role: Poster Presentation
3. Scientific Symposium Conference Current Trends in Natural Science, 2021
   - Role: Poster Presentation and Full-Length Paper Publication

Chapter 1

Introduction

1.1 Overview of Biological Complexity

Biological systems are complicated, consisting of sophisticated networks of molecular interactions that govern vital cellular operations. Human cells include between 20,000 to 25,000 genes that encode more than 100,000 proteins, facilitating a dynamic interaction among biomolecules including proteins, genes, and RNA [1]. These molecules engage inside meticulously regulated networks to sustain cellular homeostasis and govern processes such as proliferation, differentiation, and apoptosis [2]. The interaction between signaling pathways introduces additional levels of regulatory complexity.

Comprehending these molecular networks is crucial for elucidating normal physiological activities and the mechanisms that drive illnesses. Disruption in signaling pathways, especially those regulating the cell cycle and apoptosis, may result in uncontrolled cellular activities, including oncogenic proliferation [3]. Deviant gene expression, mutations in regulatory sequences, and disrupted protein-RNA interactions substantially influence disease progression. Identifying essential nodes and interactions within these networks is crucial for comprehending disease causes and formulating effective therapeutics [4]

1.2 Challenges in Data Extraction and Computational Piplines

1.2.1 Problem Definitions

A primary difficulty in bioinformatics is the effective extraction. integration, and analysis of extensive biological data from many sources. The manual recovery of data is laborious and susceptible to inaccuracies, limiting the research of intricate biological systems. This study established an automated python-based computational pipeline to facilitate the extraction, filtering, and organization of biological information for further analysis.

The data that was extracted through Python pipeline are organized in a structural relational database using MySQL, which helps to save all the downloaded data in an organized form and for easy access I can utilize it for further analysis. As I mentioned earlier the data that I extracted have various interactions including transcription factors, protein-protein interaction, gene expression patterns, and ligand-receptor interaction. Visualization was crucial to my investigation. Cyoscape was first used for network visualization, followed by the formation the python based networks on python-based tools for more specialized visualization. This scalable computational pipeline was developed to enable rapid investigation of intricate molecular networks, therefore improving understanding of their biological relevances

1.3 Goal of the Thesis

The primary aim of this thesis is to develop and validate innovative computational techniques for identifying key variables in complex biological systems. The objective of the study is to construct and assess molecular interaction networks by utilizing extensive data from specialized biological databases alongside advanced computational techniques. These networks establish a basis for pinpointing essential regulatory nodes and pathways, particularly those associated with disease processes like cancer.

The project addresses important challenges in the integration, management, and visualization of complex biological datasets. Automated computational pipelines and systematic network modeling methodologies enhance data processing and deepen the understanding of cellular regulatory networks and their changes in diseases.

1.4 Approach and Methodology

To reach the defined objective I adopted the multiphase research methodology which is summarized in the form given below:

1.4.1 Data Integration and Automation

Data from various biological databases, such as Pathway Commons, AnimalTFDB, CellTalkDB, and Genomic Data Commons (GDC), were extracted through an automated Python pipeline and integrated. The automated Python pipeline was developed by using the data specification and the database information for automated data extraction, which are listed in each database specification and documentation. By running the developed Python pipeline, the specific data of my interest are downloaded to the repository in the form of a CSV file. The downloaded data that has been processed are kept in a relational MySQL database, allowing me for scalable querying and analysis to process the data for further analysis.

1.4.2 Construction of Network and Visualization

After extracting the datasets, a network of molecular interaction networks is constructed. These networks encompass a variety of biological elements, including proteins, genes, transcription factors, and ligand-receptor interactions. To achieve effective visualization, Cytoscape was utilized for initial representations and custom Python tools were created to deliver personalized and dynamic visualizations of the networks. After a combination of connections, the interaction is visualized using Cyoscape and Python.

1.4.3 Analysis of Biological Networks

The constructed networks from the downloaded data sets were intended to pinpoint key regulatory nodes and pathways among these interactions, focusing on signal transduction and cellular processes such as transcriptional regulation. The contracted network key nodes and connections offered valuable insight into the molecular foundations of specific diseases such as cancer, focusing on how the specific nodes and interactions and pathways in these connections play a crucial role in disease progression.

1.4.4 Evaluation on Ovarian Cancer, Cell Cycle, and MAPK Pathways

This thesis presents methodologies that were validated through a case study focusing on ovarian cancer, the cell cycle, and MAPK signaling pathways. The evaluation initially centered on analyzing key transcription factors **NF-κB** and **p53**, which play crucial roles in cancer progression and cellular regulation. Using high-throughput data, the interactions of these regulators were examined to determine their influence within the network.

Following this targeted analysis, a comprehensive network was constructed by integrating large-scale datasets from online biological repositories. The network was visualized using Python and Cytoscape, facilitating a systematic investigation of molecular interactions. Various computational algorithms, including Boolean network modeling, PageRank analysis, and recurrent convolutional neural networks (RCNNs), were applied to identify key regulatory nodes within the ovarian cancer-related pathways.

The identified key nodes were further validated by comparing their roles in ovarian cancer treatment with existing literature. This approach confirmed the relevance of major regulatory elements such as **NF-κB**, **p53**, **ATM**, **TNFR1**, and other critical components involved in apoptosis and cellular signaling. These findings highlight the robustness of the computational framework in capturing essential biological interactions and identifying potential therapeutic targets.

Research Hypotheses

1. **Hypothesis 1:** An integrated computational pipeline combining Boolean modeling, PageRank, and random walk algorithms can consistently identify key regulatory nodes (e.g., NF-κB, p53, ATM) in ovarian cancer signaling networks, as evidenced by convergence across methods.
2. **Hypothesis 2:** Integrating multi-omics data (genomic, transcriptomic, proteomic) with network centrality analyses uncovers regulatory elements (e.g., IKKα, Wip1) with high centrality scores that may represent novel or underexplored targets in ovarian cancer networks.
3. **Hypothesis 3:** The proposed methodology exhibits robustness and partial generalizability, as demonstrated by its consistent identification of central nodes in both Cell Cycle and MAPK signaling pathways and by retaining $\geq$85% connectivity under random node removal.

Research Objectives

1. To construct comprehensive molecular interaction networks using integrated multi-omics data extracted from databases such as GDC, Pathway Commons, CellTalkDB, and AnimalTFDB.
2. To identify critical regulatory nodes in these networks through combined computational techniques: Boolean modeling, PageRank analysis, random walks, and RCNN-based dynamic pattern detection.
3. To validate the biological relevance of identified key nodes by comparison to established literature and known cancer-related regulators.

4. To assess the robustness of the methodology across divergent biological networks (Cell Cycle, MAPK) and under perturbation simulations.

Study Significance

This thesis contributes a validated computational framework that integrates multi-omics data with advanced network analysis tools (Boolean networks, PageRank, random walks, RCNNs) to identify key regulators in cancer signaling pathways. The work confirms the centrality of canonical regulators (NF-κB, p53, ATM) and suggests additional potential targets (IKKα, Wip1) for further experimental validation. The methodology demonstrates adaptability to other key signaling networks (Cell Cycle, MAPK) and offers a scalable approach for network-based discovery of therapeutic targets in oncology.

Thesis Statement

This thesis establishes and validates a multi-layered computational framework for the systematic identification of key regulatory nodes in complex biological networks, with an emphasis on ovarian cancer. By integrating multi-omics data with complementary computational techniques—Boolean modeling for dynamic simulation, PageRank for network topology analysis, random walks for traffic-based node importance, and RCNNs for pattern detection—the framework consistently identifies canonical regulators (e.g., NF-κB, p53, ATM) and proposes additional candidates (e.g., Wip1, IKKα) for further investigation. Validation across divergent pathways (Cell Cycle, MAPK) demonstrates the framework's robustness, adaptability, and potential for advancing systems biology-driven precision oncology.

Chapter 2

Literature Review

The literature review synthesizes historical and contemporary advances in mathematical modeling of biological systems, emphasizing the evolution of techniques for identifying key variables. From early kinetic models (Michaelis-Menten, Lotka-Volterra) to modern computational approaches (network theory, machine learning), this chapter traces the integration of engineering principles, omics technologies, and algorithmic innovations in systems biology. A critical evaluation of challenges—data heterogeneity, dynamic interactions, and multi-scale integration—frames the necessity for the methodologies proposed in this thesis. The review culminates in a discussion of how this work bridges gaps in modeling transcription factor networks and signaling pathways, particularly in cancer research.

2.1 Historical Context of Mathematical Modeling and Key Variable Identification in Biological Systems

In particular, mathematical modeling has become an essential tool for comprehending complicated biological systems in cancer research. It offers a quantitative and systematic framework that improves the investigation of cellular interactions that are essential to understanding the development of tumors. Through the use of mathematical models, scientists can express biological systems in exact mathematical formulas, enabling the quantitative examination of interactions and dynamic behaviors. Because of these models' predictive powers, researchers may simulate biological processes under different settings, predict system responses, assess scenarios, and test ideas without requiring lengthy experimental methods [5]. Mathematical formulations of empirical findings provide deep insights into the mechanics of biological events. These models aid in the identification of trends and draw attention to important details that may be missed in investigations that are only experimental [6]. Using iterative modeling

The origins of mathematical modeling in biology can be found in the middle of the 20th century, when significant advances created the foundation for understanding intricate biological systems. The Lotka-Volterra model, developed in the 1920s by Alfred Lotka and Vito Volterra, is a seminal work that used coupled differential equations to shed light on predator-prey interactions. There is ample evidence of the use of mathematical models in both physics and technology, dating back to the days of Galileo, Kepler, and Newton, the fathers of modern physics. Nowadays, modeling is considered necessary for quantitative knowledge and control in science, especially in biology. A scientific field's level of maturity is positively connected with how frequently mathematical models are created and applied to comprehend and manage real-world systems. The Lotka–Volterra model[7] for species interaction, the Hodgkin–Huxley model for neuron action potentials, the Michaelis–Menten model for enzyme-catalyzed processes, and epidemiological models for epidemics are a few prominent examples of dynamic mathematical models in biology [8].

Advances in computational biology occurred as mathematical modeling gained popularity, making it easier to construct and solve mathematical models of biological systems. The complex relationships between different organizational levels in these systems had traditionally made accurate quantitative descriptions difficult. The problem was made more difficult by the presence of open systems, several gradients that are distant from thermodynamic equilibrium, and intricate nonlinear dynamics. But in the last several years, two technological developments have completely changed the field: the birth of the "omics" sciences (genomics, transcriptomics, proteome, signalomics, and metabolomics) and the general availability of computing power. These developments have produced dynamic information about the behavior and structure of biological systems, which has simplified and reduced the cost of gathering, processing, and storing enormous volumes of data [9].

2.1.1 Emergence of Systems Biology (1950s - 1970s)

The advent of systems biology, a paradigm-shifting approach that emphasized the interconnectedness and interdependence of biological systems, in the mid-20th century signaled a turning point in the scientific community. Compared to the reductionist approaches that had typified previous molecular biology research, this approach marked a significant shift. When systems biology was first emerging, scientific research was focused on understanding the details of basic biochemical pathways. A concerted effort was made during this time to comprehend the basic molecular processes. A noteworthy effort among these was that of Michaelis and Menten in 1913, whose enzymatic reaction rates were understood thanks to the establishment of an enzyme kinetics model [10]. This early model, however initially limited to single biochemical reactions, stimulated more extensive systems-level investigation.

Michaelis and Menten's enzyme kinetics model was crucial in determining the direction that systems biology took. The concepts developed by Michaelis and Menten [10] became fundamental, influencing further research as scientists tried to broaden their reach beyond individual responses, even beyond its application to enzymatic reactions. This was a turning point where a more comprehensive understanding of biological systems was made possible by the complexities of molecular kinetics. The field of systems biology was further enhanced in the 1980s and 1990s by the incorporation of control theory and engineering ideas, building on the groundwork established by Michaelis and Menten [10]. We saw the beginning of a more quantitative and systematic approach to understanding biological systems when researchers started using concepts of control theory and system identification techniques to deduce mathematical models from experimental data. A more thorough investigation of the regulatory mechanisms governing linked systems was made possible by this integration.

When scientists realized that isolated models had their limits, a significant change happened. A substantial divergence from conventional reductionist methods was the shift from examining discrete parts to interdependent systems. The awareness that a thorough comprehension of biological phenomena necessitated taking into account the dynamic interplay between diverse parts within a system drove this paradigm shift. Scholars have adopted a more comprehensive perspective, recognizing the intricacy intrinsic to biological systems. The development of computers and other advanced analytical tools, in particular, was a major factor in the way that systems biology evolved. With the aid of these technical tools, scientists were able to handle enormous datasets and create more complex models, which allowed for a more thorough investigation of biological processes. An more nuanced knowledge of the complexity inherent in biological systems was fostered by researchers' adoption of holistic methodologies that incorporated multiple data kinds. The foundation for a more comprehensive and data-driven approach to studying life at the systems level was established by this combination of methodology and technology [11].

The 1950s to 1970s saw the rise of systems biology, which encouraged cooperation between scientists in various fields. Systems biologists adopted this multidisciplinary approach as a defining characteristic of their specialty, utilizing knowledge from engineering, physics, and mathematics to better understand biological systems. In addition to enriching the discipline, this collaborative spirit cleared the path for a more thorough and complete comprehension of the dynamic nature of living things. In conclusion, the development of systems biology in the middle of the 20th century entailed more than just improving models; it signaled a paradigm change toward holistic viewpoints, scientific breakthroughs, and interdisciplinary cooperation. Systems biology became a mainstay of modern biological study during this time, laying the groundwork for a more thorough understanding of biological systems and paving the way for future developments in the discipline. To improve our understanding of cancer biology, methods for locating important variables in intricate mathematical models of biological systems must be developed. This effort aims to improve our understanding of the dynamic connections that drive cancer biology by combining high-throughput data, network modeling, and information-flow techniques. Our goal is to improve patient outcomes for ovarian cancer and associated malignancies by laying the groundwork for future research that informs tailored therapy methods, through the validation of these methodologies and their implementations [12].

2.1.2 Integration of Control Theory and Engineering Principles (1980s - 1990s)

The fusion of engineering concepts and control theory in biological modeling during the 1980s was a revolutionary era that paved the way for the creation of techniques to pinpoint important variables in intricate mathematical models of biological systems. Researchers used system identification techniques to create mathematical models from experimental data during this era, which marked a significant paradigm shift and encouraged a more quantitative and methodical approach to understanding biological systems, especially in the context of cancer biology. The extraction of mathematical models gained popularity during this period, enabling scientists to fit theoretical frameworks to observable biological reactions in order to estimate model parameters.By moving away from merely descriptive approaches, system identification methodologies allowed for a thorough quantitative analysis, which is necessary for compiling large amounts of data and creating organized databases. For ensuing analytical activities, especially those centered on network modeling and information-flow model creation, this kind of foundation is essential [12].

Because control theory sheds light on the innate regulatory mechanisms of biological systems, it has gained new significance in the field of biology, despite its traditional engineering roots. Researchers could better understand how biological systems retain stability, react to perturbations, and display dynamic behaviors by incorporating control theory concepts. This would expand the analytical toolkit for modeling intricate interconnections in the course of cancer. This multidisciplinary approach improved comprehension and made it easier to apply engineering techniques to the biological sciences [13]. In order to create models that could both anticipate and describe observable events, control theory had to be integrated, as Angarita et al. [14] aimed to measure the impact of separate components on system behavior. This quantification of regulatory mechanisms, which made it possible to comprehend the complex networks regulating biological processes more deeply, is in line with the goals of finding the essential factors influencing cancer biology. During this time, modeling approaches evolved from solely descriptive models to more predictive and mechanistic frameworks. This change in focus inspired academics to take on the issue of modeling biological systems using concepts from control theory. Detailed biological system representations are essential for verifying new approaches in simulation and experimental investigations.

Researchers were able to investigate how systems react to both internal and external inputs when feedback and feedforward control concepts were incorporated into biological models. This allowed researchers to create a framework for developing interventions that can modify system behavior. Specifically in the field of ovarian cancer research, this application of control principles led to the development of methods for controlling biological processes, opening up new directions for therapeutic interventions and system control [15]. In conclusion, a critical step toward more quantitative and systematic methods was taken in the 1980s and 1990s with the incorporation of control theory and engineering concepts into biological models. This historical period employed mathematical methods to enhance comprehension of intricate biological systems, particularly in determining critical factors influencing the course of cancer. Control theory application has improved the modeling toolbox and led to a deeper comprehension of the regulatory mechanisms controlling these complex processes, which is directly in line with the goals of current systems biology and cancer dynamics research.

2.1.3 High-Throughput Technologies and the "-Omics" Era (Late 1990s - 2000s)

Significant developments in high-throughput technologies, such as transcriptomics, proteomics, and genomics, ushered in a revolutionary era of biological study in the late 1990s and early 2000s [16]. With the help of these breakthroughs, a paradigm shift toward the collection of large-scale data was brought about, which made it easier to create thorough genome-scale models. More potential and problems for mathematical modeling in the study of complex biological systems—particularly in the setting of cancer biology—were brought about by the growing accessibility of biological data. The speed and amount of biological data collecting were significantly increased by the introduction of high-throughput technologies. Researchers were able to study entire genomes because to methods like mass spectrometry, microarray technology, and DNA sequencing, which revealed complex patterns of gene expression and expanded our knowledge of the roles and interactions of proteins in biological processes. These domains were further advanced by advances in next-generation sequencing and better mass spectrometry, especially in the study of human disorders. Exome sequencing, for example, became a major area of interest, and whole genome resequencing became more affordable, which made it easier to find significant structural variants and advanced personalized genomic therapy [17].

Furthermore, methods like small RNA sequencing, which are crucial for locating both known and unknown microRNAs that may act as biomarkers for the diagnosis and treatment of disease, have been beneficial to transcriptomics. Targeted approaches in proteomics were essential in identifying disease-associated proteins, which are critical for clinical diagnosis and disease staging. In this age, trans-omics techniques were also integrated, where data fusion across many omics platforms was made possible by bioinformatics. This kind of integration improved diagnostic and therapeutic approaches, as well as our understanding of illness, causes [18]. There was a noticeable move toward genome-scale models as scientists adjusted to the deluge of "-omics" data, indicating a desire to embrace a more comprehensive knowledge of biological systems. The complexity of high-throughput data led to the evolution of mathematical modeling, which made it possible to create detailed models that represented the interactions and dynamics across multiple biological levels. Understanding signaling pathways and regulatory networks in biological systems depends on the study of critical variables such as ligand-receptor dynamics, transcription factors (TFs), and protein-protein interactions[19].

Cellular signaling and control are fundamentally based on protein-protein interactions. Since these interactions are essential to almost every cellular function, including signal transmission, metabolic pathways, and cellular responses to environmental changes, identifying them has become crucial. Comprehending these interplays offers perspectives on how proteins cooperate to carry out physiological tasks and how disruptions in these systems might result in illnesses, such as cancer [20]. Important regulators of gene expression, transcription factors (TFs) control the transcription of particular target genes in response to different stimuli. They frequently work by attaching themselves to regulatory elements found in the gene promoter regions, which then affects the differentiation and behavior of cells. In cancer biology, abnormal gene expression patterns that promote carcinogenesis and progression can be caused by TF dysregulation. Thus, for the development of tailored therapeutics, it is imperative to determine the interactions between TFs and their target genes [21].

The fundamental processes of cellular signaling and communication involve ligand-receptor interactions. Hormones, neurotransmitters, and growth factors are examples of ligands. Ligands attach to particular receptors on the surface of cells to start signaling cascades that affect cellular responses. Understanding these relationships and the consequences they have on different physiological processes and diseases, such as cancer, helps to explain their underlying mechanisms. Research in this area is crucial for therapeutic intervention because disruptions in ligand-receptor signaling can result in uncontrolled cell growth and survival [22]. Nevertheless, there were many difficulties in incorporating high-throughput data into mathematical models. To find important factors and correlations, new modeling methodologies were needed due to the sheer volume and complexity of "-omics" data. The integration of multi-omic datasets, noise reduction, and data standardization were among the problems that researchers had to deal with. Overcoming these obstacles became essential to gaining valuable insights from the abundance of data that high-throughput technology offered.

The "-omics" age of mathematical modeling necessitated the identification of critical variables across several biological layers. To understand how genetic alterations were reflected in the transcriptome and ultimately impacted the proteome, researchers attempted to disentangle the complex relationships that existed between genomics, transcriptomics, and proteomics. To build precise and predictive models that could represent the dynamics of intricate biological systems, it became crucial to identify these crucial variables [23]. This era was characterized by the integration of data from many "-omics" layers. Researchers took on the challenging task of combining data from transcriptomics, proteomics, and genomes to create complete models that captured the complexities of biological processes. This integration provided insights into the interactions between genes, transcripts, and proteins, enabling a more comprehensive knowledge of cellular activity.

In summary, the introduction of high-throughput technologies in the late 1990s and early 2000s caused a paradigm shift in biological modeling. Genome-scale models became popular as a result of the ability to produce enormous volumes of biological data, with mathematical modeling evolving to meet the demands of "-omics" data. By navigating the complexity to uncover critical variables across several biological layers, including transcription factors, ligand-receptor dynamics, and protein-protein interactions, researchers laid the groundwork for a more thorough and integrated knowledge of biological systems. This progression paved the way for later investigations into the intricate relationships found in biological networks, particularly in the context of cancer biology. These goals are in line with the goals of current systems biology research.

2.1.4 Rise of Computational Systems Biology (2000s - 2010s)

The field of computational systems biology emerged in the early 21st century, combining sophisticated mathematical frameworks, data-driven approaches, and computational tools to study intricate biological processes. The field of biological research saw a significant shift during this time, with a focus on modeling complex interactions within biological networks and a shift in the definition of crucial variables from isolated molecular entities to interconnected dynamics. Large-scale biological datasets produced by high-throughput technologies may be handled by researchers in the 2000s thanks to developments in algorithmic science and high-performance computing. In order to understand the intricate relationships and patterns inherent in biological systems, techniques like machine learning, network analysis, and statistical modeling have become increasingly important [24, 13]. The use of computer approaches promoted a data-centric paradigm that improved our comprehension of biological dynamics by making it easier to extract significant insights from massive amounts of data. The use of large amounts of experimental data from transcriptomics, proteomics, and genomes proved crucial to this progression. These data were used to build strong computational models, and machine learning methods were crucial in spotting complex relationships and patterns [25, 26]. Researchers were able to build predictive models that faithfully captured the intricacies of biological processes by utilizing these data-driven techniques.

Furthermore, computational systems biology became known for its sophisticated mathematical modeling, which included multi-scale interactions, feedback loops, and network dynamics. This method allowed for thorough assessments of the behavior of the system, and scientists used these models to test different theories and simulate biological processes [14]. The ability to replicate intricate biological processes in virtual settings facilitated the adjustment of parameters and the monitoring of ensuing system reactions, hence enhancing conventional experimental approaches [27, 28]. With the advancement of computational systems biology, the comprehension of essential factors was expanded to encompass network dynamics, which mirrors the interdependence of biological systems. As scientists started examining the connections between different elements in biological networks, they discovered emergent characteristics resulting from the collective actions of molecular entities [29, 30]. Deeper understanding of the complex web of biological relationships has been made possible by this emphasis on network-centric models. This has been especially helpful in discovering crucial protein-protein interactions, transcription factor dynamics, and ligand-receptor signaling pathways that control cellular behavior.

Computational models informed experimental designs and experimental data validated computational predictions, respectively, as the interaction between computational approaches and high-throughput technologies become more synergistic [31]. This mutually beneficial connection promoted a more integrative view of biological systems in addition to hastening scientific discoveries. All things considered, the decade from the 2000s to the 2010s was a turning point in biological research because of the rise of computational systems biology. Combining data-driven strategies, computer methods, and sophisticated mathematical modeling expanded the field of biological research beyond studying individual molecules to studying the intricate dynamics of biological networks. In addition to advancing the discipline of systems biology, this integrative approach well matched the objectives of current research, especially with regard to identifying critical variables impacting biological processes and disease mechanisms.

2.2 Mathematical Models in Biological Disciplines

In order to understand complicated biological phenomena, such as population dynamics, molecular interactions, and disease propagation, mathematical modeling has become an invaluable tool. In order to abstract and quantify biological processes, mathematical models are essential tools that enable researchers to forecast the behavior of systems under various circumstances. In keeping with the goals of this thesis to investigate novel modeling techniques that can capture system memory and non-linearity in biological networks, this chapter offers a thorough review of the use of mathematical models in several biological fields. A key component of understanding population and ecological dynamics has been mathematical modeling. Predator-prey interactions and species competition have been thoroughly described using classical models, such as the Lotka-Volterra equations [32]. While these models are helpful in describing the fundamental dynamics, they frequently overlook historical relationships or memory effects and assume immediate reactions. Fractional differential equations (FDEs) have been incorporated into these models in recent advances, allowing memory kernels to be included that more accurately capture the impact of previous states on current population patterns [33]. This improvement is especially pertinent to real-world situations where historical factors impact species interactions and environmental changes.

Mathematical models in the context of gene regulatory networks (GRNs) have greatly improved our knowledge of the dynamics of gene expression. The steady-state and transient dynamics of gene networks have been effectively captured by conventional models based on ordinary differential equations (ODEs) [34]. These models, however, frequently ignore the temporal delays in gene-protein interactions as well as the intrinsic stochasticity of gene expression. In response to these constraints, stochastic models and delay differential equations have been presented [35], which more faithfully capture the random variations in gene expression and the temporal lags connected to transcription and translation processes. These improved models have shed light on how cells make decisions and how resilient gene regulatory circuits are to changing environmental circumstances. The Michaelis-Menten framework, which characterizes the rate of enzymatic reactions as a function of substrate concentration, has historically been used to model enzyme kinetics, a fundamental area of biochemical research [36]. Although the traditional Michaelis-Menten equation works well for straightforward interactions between enzymes and substrates, it is unable to adequately represent the complexity of enzyme behavior in environments with multiple substrates or non-linear circumstances. To take cooperativity, temporal dependencies, and enzyme inhibition into account, recent research has expanded these models utilizing fractional kinetics and non-linear dynamics [37]. The creation of more potent inhibitors and therapeutic drugs can be aided by an understanding of the intricate dynamics of enzyme-drug interactions, which makes these improved models essential for drug development.

The significance of mathematical modeling has also been emphasized in the transmission of infectious diseases, namely in the context of the COVID-19 pandemic. To simulate disease transmission, forecast epidemic peaks, and assess the effectiveness of intervention methods, traditional epidemiological models like the Susceptible-Infected-Recovered (SIR) framework have been used [38]. However, these models' limited relevance to real-world circumstances stems from their basic character, which assumes homogeneous mixing and constant transmission rates. Recent models have used fractional calculus to add memory effects, network-based interactions, and time-dependent transmission rates to get around these restrictions [39]. These adjustments enable a more realistic depiction of the dynamics of disease and offer improved direction for choices in public health policy. Since the introduction of deep learning-based methods like AlphaFold, protein folding models in molecular biology have significantly changed [40]. Machine learning algorithms have outperformed traditional models that relied on heuristic search techniques and simplified energy landscapes to predict protein structures with nearly experimental accuracy. This discovery has fundamentally changed our knowledge of the stability and function of proteins, with ramifications for synthetic biology and medication development. Another area where mathematical approaches have had a significant impact is metabolic pathway modeling. To predict biological reactions to environmental changes and simulate the interactions between different biochemical pathways, genome-scale metabolic models (GEMs) have combined multi-omics data [41]. These models have been used to investigate treatment targets, find possible biomarkers, and investigate cancer metabolism. These models are now much more predictive thanks to the addition of constraint-based optimization techniques like flux balance analysis (FBA), which allows scientists to simulate cell development and metabolism under various genetic and environmental scenarios. The action of interconnected neurons is simulated using neural network models, which are incredibly useful for studying brain activity and cognitive functions. From straightforward abstractions of neuron dynamics, these models have developed into complex simulations that accurately represent the complex architecture and operation of entire brain regions [42]. These models have become more accurate via the addition of biologically plausible factors like neurotransmitter dynamics and synaptic plasticity. This makes them valuable resources for comprehending neurological illnesses and creating brain-machine interfaces.

Drug development greatly benefits from the use of pharmacokinetic and pharmacodynamic (PK/PD) models, which simulate the absorption, distribution, metabolization, and excretion of medications (PK) as well as their effects on the body (PD) [43]. Historically, these models have been based on compartmental techniques; however, new developments have integrated the concepts of systems biology to represent the intricate relationships that exist between pharmaceuticals and biological systems across various dimensions. More thorough pharmacological action simulations are now possible because to the incorporation of quantitative systems pharmacology (QSP). These simulations are essential for maximizing beneficial effects and adjusting dosage schedules. Finally, mathematical modeling offers us a strong lens through which to examine and comprehend the intricacies of biological systems. These models provide a structured framework for analyzing the nonlinear, stochastic, dynamic nature of biological interactions by connecting theoretical concepts with empirical findings. By combining cutting-edge modeling methods with experimental data—like fractional calculus and machine learning—we want to progress biological research and address more complicated issues while creating more potent treatment plans.

2.3 Importance of Key Variables in Mathematical Modeling for Systems Biology in Biomedical Engineering

Finding key variables in mathematical models is essential to comprehending and predicting the intricate behaviors of biological systems in biomedical engineering. This section emphasizes the importance of these factors in a variety of applications, with a particular emphasis on how they affect biological outcomes and modeling accuracy. In biomolecular processes, critical variables like molecular concentrations, binding affinities, and reaction rates are crucial. To simulate biochemical networks successfully, these factors must be accurately identified and quantified. For example, Michaelis-Menten kinetics are directly influenced by substrate concentration and enzyme affinity, which impacts the prediction power of metabolic rates [44]. Nonetheless, difficulties arise from inconsistent experimental results and a poor comprehension of system components. The accuracy and predictive capability of the model are improved by sophisticated techniques including parameter estimation and sensitivity analysis [45].

Important elements like pharmacokinetics (absorption, distribution, metabolism, and excretion) and pharmacodynamics (drug-receptor binding) are necessary for precise modeling in the field of drug interactions with biological systems. These characteristics have a major impact on dosing techniques and the efficacy of therapy [43]. It is still difficult to integrate multi-scale data from the molecular to the organismal levels. Drug activity predictions in complicated biological systems can be improved by using hybrid models that include deterministic and stochastic techniques [46]. Implant efficacy is determined by biomaterials design characteristics such as mechanical qualities and degradation rates. It is difficult to comprehend how biomaterials and biological tissues interact, but optimization for particular applications is possible when computational modeling and experimental validation are combined [47].

Signaling networks are complex systems that require advanced modeling methods. A richer knowledge of biological processes is made possible by systems biology frameworks that integrate omics data to highlight the roles of important variables [48]. Genetic polymorphisms and metabolic rates are two patient-specific characteristics that must be taken into account in personalized treatment. Individual responses to medicines can be predicted by customized models, which optimizes treatment plans [49]. For the purpose of recreating disease trajectories and improving predictions of disease progression, critical variables including immune response dynamics and cancer cell growth rates are essential [50]. Using probabilistic modeling and Bayesian inference, one may improve prediction accuracy by tackling biological heterogeneities and uncertainties through frameworks [51]. In bioprocess engineering, temperature, nutrient contents, and pH all have an impact on biotherapeutic yield. Production process optimization requires adaptive control and real-time monitoring [52]. Finding important factors makes it possible to assess the resilience of biological systems and create control plans that ensure desired outcomes [53]. Achieving precision and understanding in mathematical modeling in biomedical engineering requires the identification of key variables. It makes it easier to comprehend biological systems, optimize therapies, and create novel answers to challenging biomedical problems.

2.4 Challenges in Identifying Key Variables for Systems Biology and Engineering in Biomedical Engineering

Because biological systems are inherently complex and engineering principles must be integrated, identifying essential variables in systems biology and biomedical engineering poses several obstacles. The interdisciplinary synergies necessary for successful variable identification are frequently missed by traditional techniques [54]. Utilizing bioinformatics tools and multi-scale modeling, advanced systems biology approaches improve the identification of critical elements governing complex biological processes [55]. Dynamic interactions at different sizes and timescales define biological systems. Important interactions and feedback loops that are required for precise predictions are frequently missed by static models [56]. The temporal changes and interactions occurring in biological systems can be represented using dynamic systems modeling, which includes agent-based and time-series analyses [57].

Integrating quantitative data (genomic sequences, for example) with qualitative data (clinical observations, for example) is essential to a thorough understanding of biological systems. Harmonizing these disparate data formats is a challenge for traditional approaches [58]. In order to improve the identification of crucial components, probabilistic graphical models are used in hybrid modeling approaches, which combine mechanistic models with machine learning techniques [59]. Heterogeneity characterizes the landscape of biological and clinical data, which includes a variety of datasets including proteomic, imaging, and genomic data. In order to generate a coherent representation of biological systems and make it easier to identify critical elements, multi-omics techniques and data integration systems combine various data types using techniques like data fusion and meta-analysis [60].

Understanding the factors that contribute to resilience and stability is crucial for grasping system robustness. However, traditional models often overlook intricate feedback processes and control mechanisms, limiting our insight into how systems respond to various disturbances. The discovery of variables that increase stability is made easier by resilience engineering and robust control design approaches, which enhance knowledge of system reactivity under uncertain conditions [61]. The need for interdisciplinary approaches is further highlighted by the identification of important variables in systems biology and engineering. The construction of models that faithfully capture the complex inner workings of biological systems is made possible by the bridging of mathematical modeling with biological insights [62].

2.5 Bridging Mathematical Modeling and Biology

Biological processes must be understood using mathematical modeling, which provides a methodical way to examine dynamic interactions in living systems. These models combine mathematical rigor with biological complexity, providing insights that guide experimental design and technological applications [63]. They are built utilizing computational algorithms, stochastic processes, and differential equations. Gene regulatory network models, for instance, employ differential equations to make sense of the relationships between genes, proteins, and metabolites, providing insights into cellular processes and disease mechanisms [64]. Furthermore, population dynamics models improve our comprehension of how evolution has adapted in ecological settings. To handle the complexity of biological systems, cooperation between mathematicians and biologists is essential. Biologists verify that the models developed and analyzed by mathematicians appropriately reflect biological reality. Mathematicsians design and evaluate models using quantitative methodologies. In complicated phenomena like disease progression, where biological facts support mathematical predictions, this interdisciplinary alliance creates models that are both mathematically valid and biologically meaningful [65, 66].

This work uses basic mathematical and statistical principles to examine intricate biological systems. Foundational tools include differential equations, such as partial differential equations (PDEs) for spatial phenomena like tissue diffusion and ordinary differential equations (ODEs) for population growth and enzyme kinetics [67]. ODEs play a critical role in epidemiology as well, allowing estimates of the spread of infectious diseases and the effectiveness of treatments [68]. Statistical intelligibility and probability are essential for handling uncertainty in biological systems. While statistical techniques make parameter estimation and model validation easier, stochastic models represent the randomness in processes such as gene expression [69]. By continuously updating models with fresh data, Bayesian techniques improve forecast accuracy[70]. Linear algebra facilitates the representation of biological interactions using matrices and vectors, and eigenvalues and eigenvectors can be used to infer system stability [71]. Systems biology has demonstrated the efficacy of eigenvalue analysis in identifying key regulatory genes affecting biological activities [72]. This strategy is especially pertinent to the drug research industry, where algorithms are used to optimize treatment efficacy and minimize side effects. To find the optimal conditions, optimization theory further refines these models [73, 74].

Biological networks are commonly modeled using graph theory, which employs nodes to represent biological entities and edges to indicate relationships. This method aids in deciphering network topologies and locating crucial elements [75]. Finding significant metabolites and possible treatment targets inside metabolic networks is made possible by this kind of study [76]. Using methods such as autocorrelation and Fourier analysis, time series analysis exposes patterns in dynamic biological processes [77]. As demonstrated in circadian biology, these models are especially helpful in understanding the temporal regulation of gene expression [78].

By revealing nonlinear correlations, machine learning techniques like neural networks are very helpful in analyzing complex biological information [79]. By predicting gene functions and regulatory elements from sequence data, these methods are advancing genomic research [80]. The biological complexity has been substantially improved by the identification of key variables in mathematical models. Single-cell RNA sequencing exposes cellular heterogeneity, whereas multi-omics integration integrates data from multiple sources to offer a comprehensive picture of biological systems [81, 82]. By utilizing machine learning and deep learning techniques to improve the identification of critical variables, Dynamic Bayesian Networks (DBNs) mimic time-dependent processes and improve predictive accuracy [83, 84]. Furthermore, methods for concluding causality, like causal graphical models, aid in distinguishing between correlation and causation, enhancing the accuracy of the variables that have been found [85].

Topological Data Analysis (TDA) facilitates the identification of key components by showing the connection and structure of biological networks [86]. Collaborative platforms and open data efforts aelerate research by promoting data accessibility and sharing [87]. However, interpretability problems persist, necessitating the use of explainable AI techniques to clarify model selections and increase trust in findings [88]. Finding the important variables in mathematical models has broad ramifications for many different sectors. Comprehending the fundamental reasons behind illnesses facilitates the creation of customized therapies and diagnostic instruments in the field of medicine [89]. In medication development, this method lowers expenses while increasing efficacy [90]. While precision agriculture employs these insights to optimize methods for higher crop yields and resource efficiency, environmental studies apply the same ideas to conservation efforts and sustainable resource management [91].

These understandings are crucial for public health decision-making during epidemics, vaccination strategy optimization, and resource allocation [92]. Increasing manufacturing process optimization boosts efficiency in bioprocess engineering [5]. In addition, strong decision support systems employ these factors in computer models to develop informed healthcare and environmental policy [93]. Finding essential elements in synthetic biology propels developments in the creation of biological systems with clear roles, advancing the field of synthetic circuits and genetic networks[94]. These uses highlight how mathematical representations of biological systems have the power to transform industries including biotechnology, environmental research, medicine, and agriculture.

2.5.1 Summary and Transition to Methodology

To systematically investigate key regulatory variables in cancer-related networks, we initially constructed a prior network integrating interactions between the **NF-κB** and **p53** pathways form pathway common datasets. This foundational network was enriched with weighted data derived from ovarian cancer clinical datasets obtained from The Cancer Genome Atlas (TCGA), ensuring that both biological interactions and clinical relevance were incorporated.

To systematically identify critical nodes within the constructed NF-κB and p53 interaction network, we applied multiple analytical techniques, each providing distinct insights into node importance and regulatory significance:

- **Random Walk:** A stochastic process was used to simulate the probability of reaching a node based on network connectivity. Nodes with higher probabilities were considered more influential in the signaling cascade. **Input:** Weighted gene interaction network derived from ovarian cancer datasets. **Output:** Probability scores for each node, highlighting key regulators.

- **PageRank:** This algorithm ranked nodes based on their global significance within the network, emphasizing nodes with a high number of influential connections. It provided a measure of hierarchical importance in regulatory pathways. **Input:** Directed gene interaction network with assigned weights. **Output:** A ranked list of nodes indicating their centrality within the network.

- **Boolean Modeling:** Regulatory interactions were analyzed through discrete logical modeling, capturing the activation or inhibition of genes under specific conditions. This method helped in identifying key regulatory switches in the system. **Input:** Binary representation of regulatory interactions. **Output:** Functional classification of genes based on their activation states.

- **Recurrent Convolutional Neural Networks (RCNN):** Deep learning techniques were employed to detect intricate patterns within the interaction network, improving the understanding of how genes influence each other dynamically. **Input:** Feature-encoded network representation with weighted interactions. **Output:** Predictive scores indicating significant regulatory interactions and potential driver genes.

These methods collectively facilitated the identification of key regulatory elements by ranking network components based on their importance in oncogenic processes. To streamline the analysis, I developed an automated pipeline using Python scripts to extract interactions from publicly available datasets. A structured database was designed to store and manage extracted information efficiently. The processed data was filtered based on biologically significant interactions and formatted into a network structure, initially visualized using Cytoscape before being transferred back to Python for further computational analysis. To validate the efficiency and robustness of the proposed methodology, I extended the analysis beyond ovarian cancer by applying the pipeline to two additional networks: the **Cell Cycle** and **MAPK** signaling pathways. The entire workflow, including data extraction, network construction, ranking algorithms, and visualization, was systematically repeated for these networks to ensure the generalizability of our approach. The following chapter details the methodological framework implemented in this study, describing each step of data processing, network modeling, and analytical validation in a structured manner.

Conclusion of the Chapter

This review underscores the transformative impact of computational tools in deciphering biological complexity, from enzyme kinetics to genome-scale networks. The synthesis of historical context and cutting-edge techniques highlights the unmet need for scalable, automated pipelines to analyze regulatory networks. By building on these foundations, the present study addresses critical limitations in data integration and dynamic modeling, paving the way for advancements in therapeutic target identification. The subsequent methodology chapter will detail the technical implementation of these principles.

Chapter 3

Methodology

The methodology chapter outlines the systematic approach employed to identify key variables in complex biological networks, with a focus on ovarian cancer pathways. This study integrates multi-omics data from high-throughput databases such as *Pathway Commons*, *AnimalTFDB*, and the *Genomics Data Commons (GDC)* to construct a comprehensive computational pipeline. The methodology is divided into four phases: (1) automated data extraction and integration, (2) network construction and visualization, (3) advanced computational analysis using Boolean modeling, PageRank, random walks, and RCNNs, and (4) validation across additional pathways (*Cell Cycle* and *MAPK*). Each phase is designed to ensure robustness, scalability, and biological relevance, providing a framework for reproducible analysis of regulatory networks in systems biology.

3.1 Purpose of the Study

The primary purpose of this research project is to develop an integrated network model that can understand the key connections between biological entities in ovarian cancer. By constructing a reliable database, merging high-throughput genomics data, and employing powerful computational approaches, this study aims to identify critical receptor-transcription factor interactions and disclose fundamental variables impacting cancer biology.

This chapter shows the methodical procedure for collecting, processing, and analyzing data. It describes how to create network models, analyze relationships, and design integrated databases. The pseudocode provides full procedural guidance for the technique.

Table 3.1: Consolidated Pseudocode for Methodology

Step	**Description**
Step 1: Data Collection and Integration	
- **Genomics Data Collection**	Collect high-throughput genomics data from public repositories, such as the Genomics Data Commons.
- **Transcription Factor Data Collection**	Gather transcription factor data from AnimalTFDB 3.0 to identify key regulators.
- **Pathway Data Collection**	Collect pathway information from Pathway Commons integrated with KEGG.
- **Ligand-Receptor Data Collection**	Gather data on ligands and receptors from the Cell Talk Database to model receptor-ligand interactions.
Step 2: Data Processing and Integration	
- **Data Preprocessing**	Clean and preprocess collected data to remove inconsistencies and missing values.

Continued on the next page...

Step	Description
- **Network Construction**	Construct a directed network graph incorporating the processed data, with nodes representing biological entities and edges representing interactions.
- **Database Design**	Design a structured database schema to store and efficiently retrieve integrated biological data.
Step 3: Advanced Interaction Analysis	
- **Gene Interaction Analysis**	Analyze gene-gene interactions to uncover regulatory dependencies.
- **Protein-Protein Interaction Analysis**	Identify and assess protein-protein interactions critical for signaling processes.
- **Pathway Crosstalk Analysis**	Examine pathway crosstalk to understand interplay between different signaling pathways.
- **Regulatory RNA Interaction Analysis**	Analyze the interactions of regulatory RNAs, such as microRNAs, within the network.
- **Transcriptional Network Analysis**	Investigate interactions between transcription factors and their target genes.
- **Functional Process Interaction Analysis**	Evaluate interactions involved in specific functional biological processes, such as apoptosis or proliferation.
- **Genomic Region Interaction Analysis**	Analyze interactions specific to defined genomic regions.
Step 4: Database Structure Implementation	
Database Schema Implementation	Implement the designed schema to organize and store the integrated data effectively.
Data Insertion	Insert genomics, transcription factor, and pathway data into the database using predefined formats.
Data Retrieval and Querying	Develop efficient queries for data retrieval and perform initial data analysis to validate structure.
Step 5: Data Processing and Integration	
High-Throughput Data Processing	Preprocess RNA sequencing and whole-genome sequencing data from GDC, including quality control, normalization, and differential expression analysis to identify significant genes.
Signaling Pathway Data Integration	Extract and format signaling pathways from Pathway Commons and integrate with high-throughput data for comprehensive network analysis.
Step 6: Network Modeling	
Initial Network Construction	Build a basic network model using signaling pathway data from databases like Pathway Commons.
Integration of High-Throughput Data	Incorporate processed and normalized RNA-seq and WGS data to predict gene linkages and interaction intensities using correlation analysis.
Validation and Improvements	Compare predicted networks to established models and iteratively refine the network inference method to enhance accuracy and reliability.
Step 7: Signal Flow Modeling	
Random Walk Algorithm	Simulate the movement of a "walker" across the network based on edge weights, uncovering node significance, patterns, and network topology by analyzing node visitation frequencies and edge traversal patterns.
PageRank Algorithm	Rank nodes based on their importance in the network to identify critical regulatory genes or proteins, considering both local and global network structures.

Continued on the next page...

Step	Description
Degree Centrality	Measure the number of direct connections a node has to understand its local importance in the network.
Eigenvector Centrality	Identify globally influential nodes by considering both the number and the significance of the connected nodes.
Betweenness Centrality	Determine nodes acting as bridges or bottlenecks in the network by measuring the frequency of a node lying on the shortest path between others.
Step 8: Network Construction and Simulation	
Network Construction	Construct a directed graph with nodes representing molecular entities and edges denoting regulatory interactions, using NetworkX in Python.
Data Sources	Use data from Genomics Data Commons (GDC), Pathway Commons, CellTalk, and AnimalTFDB to define graph topology.
Boolean Modeling of Network Dynamics	Simulate network behavior using Boolean models to identify pivotal nodes and stable states.
Step 9: RCNN Model for Network Analysis	
RCNN Model Architecture	Implement an RCNN model using TensorFlow and Keras to dynamically analyze signal propagation, identifying critical nodes.
Graph Representation and Data Preparation	Prepare an adjacency matrix of the network and feature vectors based on node properties such as centrality and clustering coefficients.
Training and Optimization	Train the RCNN model using the Adam optimizer, employing data augmentation and early stopping to prevent overfitting.
Step 10: Methodology for Additional Network Tests	
Network Construction and Modification	Construct additional network configurations by modifying nodes and edges while maintaining the overall structure. Replace selected nodes with functionally analogous ones and redefine edges to represent alternative biological interactions.
Validation Across Modified Networks	Apply the same analytical methods (e.g., Random Walk, Boolean Modeling, PageRank, Centrality Measures) to evaluate node significance in modified networks, ensuring consistency and robustness.
Comparative Analysis	Conduct a comparative analysis to evaluate overlaps between modified and original networks, reinforcing the universal applicability of the methodology.

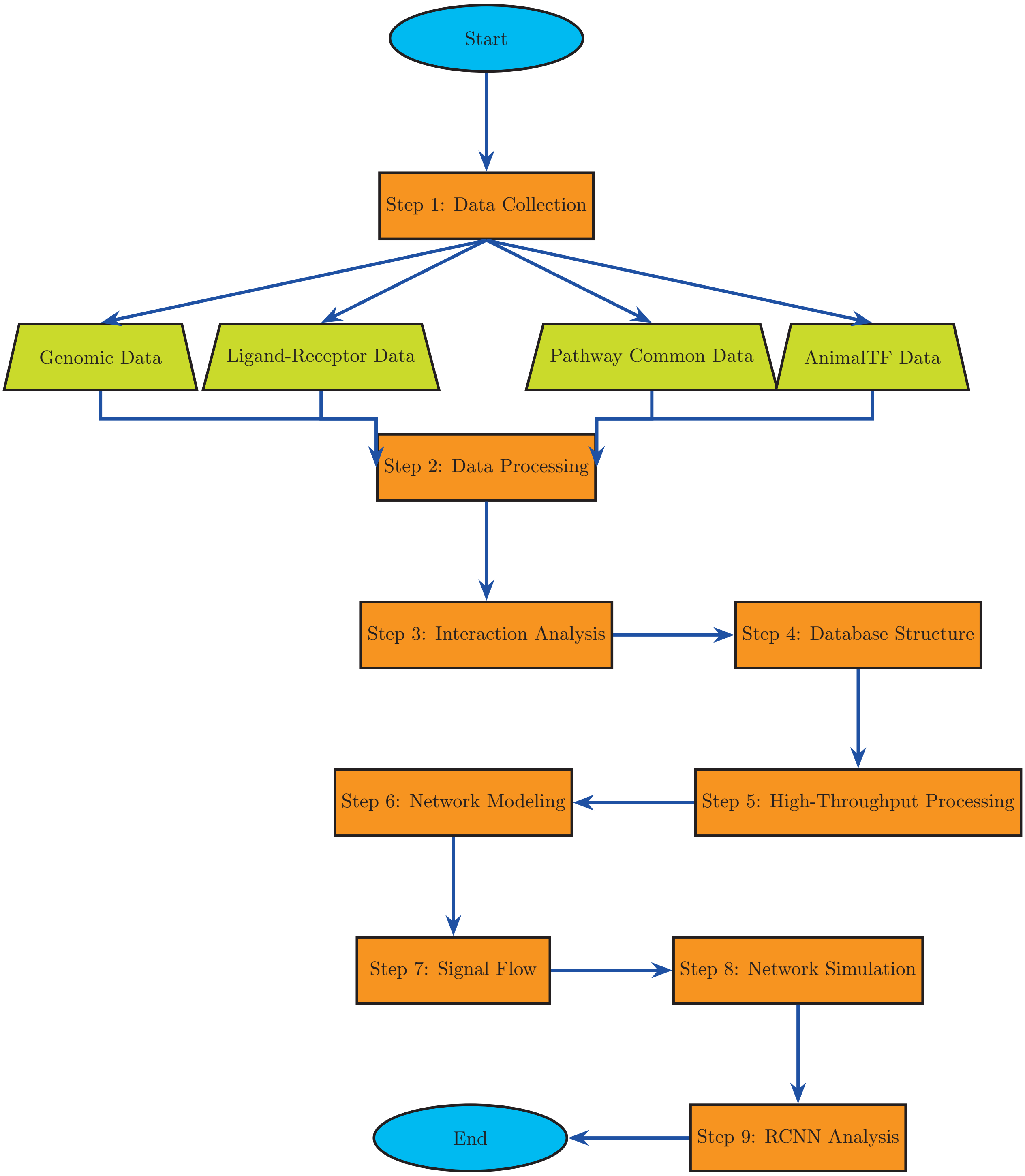


Figure 1: Flowchart Representing the Methodology

3.2 Data Collection Overview

A pivotal purpose of this study was data collection, which comprised a range of sources and formats to properly address each research issue. The stepwise procedure for data collection and operations is described in detail below.

3.2.1 Detailed Data Analysis

Each stage of the study was meticulously designed to directly address the research goals. From data collection to network formation, each strategy provided novel insights into the interactive association between the biological systems being studied.

3.2.2 Network Modeling

A static network was constructed using data-driven network inference techniques. The methodology consists of the following steps:

1. **Initial Network Construction:** A draft network was created using signaling pathway data from databases such as Pathway Commons. This served as the foundational structure for further refinement.

2. **Data Extraction:** High-throughput data, including RNA sequencing and whole-genome sequencing data, were obtained from the Genomics Data Commons (GDC) web portal. The data were processed, standardized, and integrated into the network.

3. **Network Inference:** Correlation analysis was applied to infer gene linkages and quantify interaction strengths. This step integrated the extracted data into the draft network, enhancing its accuracy and predictive power.

4. **Network Validation:** The inferred network was validated through iterative comparison with established biological network models (e.g., p53, NFkB, and ATM). This process involved:

    - **Topological Consistency:** Assessing structural properties such as node degree distribution, clustering coefficients, and connectivity patterns to ensure alignment with known biological networks.

    - **Biological Relevance:** Identifying overlapping nodes and edges between the inferred network and reference models to evaluate functional consistency.

    - **Novel Element Detection:** Detecting and validating novel elements (nodes or edges) not present in the reference models. These elements were cross-referenced with literature or experimental data to confirm their biological significance.

    In cases where the inferred network differed in size (e.g., more or fewer nodes) from reference models, the following steps were taken:

    - **Biological Plausibility Check:** Additional nodes or edges were evaluated for their biological plausibility based on existing knowledge or experimental evidence.

    - **Refinement of Inference:** The network inference process was iteratively refined to minimize false positives (incorrectly inferred interactions) and false negatives (missed interactions).

The proposed methodology was validated using well-known biological network models (e.g., p53, NFkB, and ATM). The validation process ensured that the inferred network was both consistent with established knowledge and capable of identifying novel, biologically relevant elements. This approach ensures robustness and generalizability, making the methodology applicable to a wide range of biological systems..

3.2.3 Data Collection and Integration

Genomics Data Commons (GDC)

This work collected and extracted clinical and genomic data on ovarian cancer patients from the Genomics Data Commons (GDC) database using a systematic technique. The technique's details are in the following subsections. A custom crafted Python script was created to facilitate the collection and integration of data. The following outlines the process's primary phases, and the extra resources offer the entire Python script. A, table A.4.

Accessing the GDC Database

- The first step involves accessing the database by using the GDC database API, accessible at `https://api.gdc.cancer.gov`. The GDC database is a rich source of clinical and genetic data, with a strong emphasis on individuals with ovarian cancer.

Requesting File IDs

- To apply filters and get file IDs for the pertinent data kept in the GDC database, a custom Python script was used.

Initiating Data File Download

- The identified data files were downloaded initially by the script. The full script is included in the supplemental tables.

Utilizing Custom Download Function

- The script included a custom function, `custom_download_file`, to effectively handle the download procedure for each file ID, ensuring successful retrieval of ovarian cancer files from the GDC database.

Generating Download URL

- The download URL was dynamically created by appending the file UUID and selecting CSV as the format.

Downloading Data in Chunks

- The files were downloaded in segments using the `requests` library and saved locally in the designated destination directory.

Implementing Error Handling

- Error-handling procedure, including retries mechanism and pause periods, were included in the script to ensure successful retrieval and manage issues during the download process.

Processing Downloaded Data

- The downloaded CSV files now contain the essential high-throughput data specific to ovarian cancer, ready for seamless integration into the analysis pipeline.

Further Analysis and Integration

- For additional analysis and integration with relevant clinical data specific to ovarian cancer, the files can be processed using data manipulation tools such as `pandas` or loaded into an appropriate database.

In summary, this methodology effectively facilitated the extraction of necessary high-throughput data related to ovarian cancer from the GDC database Table 3.2 is given below, mentioning all the details, ensuring a streamlined process for its integration into subsequent analyses.

3.2.4 Pathway Commons Database

This study utilized the Pathway Commons database to collect detailed information on various biological pathways and interactions. To achieve comprehensive pathway coverage, we accessed the KEGG (Kyoto Encyclopedia of Genes and Genomes) resources within the Pathway Commons database. The following steps outline the procedure, with the complete Python script detailed in A, table A.1:

1. **Data Retrieval:** Extract data from the Pathway Commons database, which contains detailed information on biological pathways and molecular interactions.
2. **Utilization of KEGG:** Leverage the KEGG component of the Pathway Commons database to obtain detailed pathway information.

Data extraction was performed using a Python script that incorporated several packages, including `gzip` for file decompression, `urllib.request` for database URL access, `pandas` for data processing, and `contextlib` for resource management.

Table 3.2: Data Extraction Steps from Genomics Data Commons (GDC)

Step	Description
Access GDC Portal	Establish connection to the Genomics Data Commons (GDC) portal to initiate data retrieval.
Query File IDs	Request specific file IDs corresponding to genomic data of interest from the GDC repository.
Initiate Data Download	Start downloading selected data files from the GDC using authorized access credentials.
Custom Download Function	Utilize a customized Python function to manage and optimize the download process, ensuring efficient handling of large datasets.
Generate Download URLs	Dynamically generate URLs based on file IDs and authentication tokens to facilitate secure and authenticated data downloads.
Download Data in Chunks	Retrieve data in manageable chunks to optimize download speed and minimize network congestion during transfer.
Implement Error Handling	Implement robust error handling mechanisms to manage network disruptions or incomplete downloads, ensuring data integrity.
Process Downloaded Data	Parse and process downloaded data files to extract relevant genomic information, and convert formats if necessary for further analysis.
Analyze and Integrate Data	Perform comprehensive data analysis, including statistical analysis, visualization, and integration with existing datasets to derive biological insights.

The primary function, `DownloadAndProcessPathwayCommonsData`, manages the downloading and processing of pathway data. This function accepts an optional argument, `filename`, which specifies the file path and name for storing the extracted data.

The function operates as follows:

1. **Access Data:** Utilize `urllib.request` and `gzip.GzipFile` to access and decompress the file from the specified URL.
2. **Read Data:** Employ `pd.read_csv` from `pandas` to read the file in chunks of 1000 rows for efficient handling.
3. **Filter Data:** Filter the dataset to include only rows with interaction types specified in the predefined `interactions` list.
4. **Remove Column:** Eliminate the "MEDIATOR_IDS" column, which is deemed unnecessary for subsequent analysis.
5. **Format Conversion:** Convert the filtered data into a nested list structure to facilitate further processing.
6. **Define Columns:** Specify column names as `["PARTICIPANT_A", "INTERACTION_TYPE", "PARTICIPANT_B", "INTERACTION_DATA_SOURCE", "INTERACTION_PUBMED_ID", "PATHWAY_NAMES"]` for clarity and organization.
7. **Create DataFrame:** Transform the nested list into a `pandas` DataFrame using `pd.DataFrame`, enabling structured data manipulation.
8. **Save Data:** Save the processed DataFrame to a CSV file using the specified filename, ensuring data persistence and future accessibility.

Upon successful data extraction, the function outputs the message "data downloaded." When executed, the script will display results indicating the completion of the data retrieval process. The Python script employed for this extraction operation is detailed in the supplementary materials.

This methodology outlines the systematic approach employed for acquiring access to the Pathway Commons database, utilizing the Kyoto Encyclopedia of Genes and Genomes (KEGG) to extract essential signaling pathway data, and subsequently saving the data as a CSV file for further analysis. The steps undertaken during this process are detailed in Table 3.3.

The extraction of signaling pathway data from the Pathway Commons database involves several sequential steps designed to ensure a comprehensive and reliable dataset.

Table 3.3: Data Extraction Steps from Pathway Commons Database

Step	**Action**	**Detailed Description**
Access DB	Connect	Establish a connection to the Pathway Commons database utilizing appropriate API endpoints.
Retrieve Data	Fetch	Employ queries to retrieve relevant signaling pathway data based on specified parameters.
Utilize KEGG	Employ	Integrate KEGG resources within Pathway Commons for comprehensive pathway details and interactions.
Load Libraries	Import	Import essential Python libraries: `pandas`, `contextlib`, `urllib.request`, and `gzip` for data handling.
Define Function	Create	Develop the function `DownloadAndProcessPathwayCommonsData` to encapsulate the data extraction process.
Specify URL	Provide	Define the specific database URL and relevant interaction types required for data retrieval.
Extract File	Download	Download the data file and decompress it using the `urllib.request` and `gzip` libraries.
Read Chunks	Load	Load the data incrementally in chunks of 1000 rows to optimize memory usage and processing efficiency.
Filter Data	Apply	Filter the dataset based on predefined interaction types relevant to the study.
Clean Data	Remove	Eliminate extraneous columns, such as `MEDIATOR_IDS`, to streamline the dataset.
Organize Data	Convert	Convert the filtered data into a nested list format for structured analysis.
Define Columns	Specify	Clearly define column names in the dataset to enhance readability and usability.
Create DataFrame	Transform	Transform the nested list into a `pandas` DataFrame for advanced data manipulation and analysis.
Save CSV	Persist	Save the processed DataFrame as a CSV file with an appropriate filename for future reference.
Confirm Completion	Provide	Output a confirmation message to indicate successful data extraction and processing.

3.2.5 Data Extraction from CellTalk Database

In parallel to the data acquisition from the Pathway Commons database, a focused effort was made to explore intercellular communication pathways through data extracted from the CellTalk database. The steps outlined below detail this extraction methodology, with the comprehensive Python script provided in the supplementary materials (refer to Table A.2).

1. **Loading Essential Packages:** Essential Python packages, such as `pandas` and `contextlib`, were loaded to facilitate efficient data manipulation and management.
2. **Defining the Function:** A function named `DownloadAndProcessCellTalkData` was defined to streamline the extraction process. This function not only downloads the data but also processes it, allowing users to specify the filename and storage path.
3. **Downloading Data:** Utilizing the `urllib.request` package, the function accessed the CellTalk

database using a specified URL. The `pandas.read_csv` method was employed to optimize data retrieval by fetching 1000 rows in batches, ensuring efficient memory usage.

4. **Data Handling:** Upon successful retrieval, the dataset underwent meticulous processing. The raw data was systematically organized into a structured format, enhancing coherence and accessibility. Comprehensive column names were developed to improve clarity and consistency throughout the dataset.
5. **Protecting Data Integrity:** To preserve data integrity for future research applications, the processed dataset was saved as a CSV file. Users were allowed to specify the filename and directory upon executing the function, facilitating seamless integration with other datasets for comprehensive analysis.

The successful extraction from the CellTalk database yielded critical insights into ligand-receptor interactions, which are vital for understanding intercellular communication mechanisms. All details are in the given table 3.4. This foundational knowledge supports further investigation into the intricate biological processes underpinning cellular dynamics.

Table 3.4: Data Extraction Process from CellTalk Database

Name	**Action**	**Details**
Load Packages	Loading Python packages	Load necessary Python packages: `pandas`, `contextlib`, etc.
Define Function	Function definition	Define the `DownloadAndProcessCellTalkData` function for efficient data processing.
Access Database	Database access	Connect to the CellTalk database via the specified URL using `urllib.request`. Read data in chunks of 1000 rows using `pandas.read_csv`.
Organize Data	Data organization	Flatten and structure the data into a nested list format for further analysis.
Define Columns	Column definition	Specify clear and concise column names for enhanced clarity: `Ligand_NAME`, `Receptor_TYPE`, etc.
Save Data	Data saving	Save the processed data as a CSV file, allowing the user to specify the filename and path.

3.2.6 Data Acquisition from AnimalTFDB

To obtain data from the Animal Transcription Factor Database (AnimalTFDB), which encompasses a comprehensive dataset describing transcription factors across various species, we implemented a methodical approach. This involved the careful curation of transcription factors specifically related to the human genome. The detailed Python script utilized for data extraction is provided in the supplementary materials (refer to Table A.3).

The following steps summarize the methodology for downloading and processing AnimalTF data:

1. **Import Required Libraries:** Essential libraries were imported, including:
   - `pandas` for data manipulation and analysis.
   - `contextlib` for managing resources effectively.
   - `urllib.request` for downloading files from the internet.
   - `gzip` for handling compressed data files.
2. **Define the Extraction Function:** A function named `DownloadAndProcessAnimalTFData` was defined. This function is responsible for downloading AnimalTF data from a specified URL and processing it into a user-friendly format.
3. **Specify Function Arguments:** The function accepts one argument:

- `filename` — The path where the downloaded file will be saved, defaulting to `"AnimalTFData.csv"`.

4. **Download Data:** The `urllib.request` library was employed to download the AnimalTF data from the provided URL. The downloaded file is handled as a compressed `.gz` file to facilitate efficient storage and transfer.

5. **Read Data:** The downloaded data is read in increments of 1000 rows to manage large datasets effectively, followed by flattening and restructuring it as a list of rows.

6. **Convert to DataFrame:** The list of rows is then converted into a `pandas` DataFrame, facilitating straightforward manipulation and analysis.

7. **Save Data:** The final DataFrame is saved to a CSV file using the specified `filename` to ensure data integrity and ease of access.

8. **Output Message:** Upon completion, the function outputs a message indicating successful download and processing of the data.

9. **Main Execution:** The extraction function is executed, invoking the download and processing of the AnimalTF data. The user can easily run this in their Python environment.

This methodical approach to data acquisition from the AnimalTFDB enables a comprehensive understanding of transcription factors and their roles in various biological processes all details of the steps involve are in table 3.5.

Table 3.5: Data Extraction Process from AnimalTFDB

Name	**Action**	**Details**
Import Libraries	Load libraries	Import `pandas`, `contextlib`, `urllib.request`, and `gzip`.
Define Function	Function definition	Define `DownloadAndProcessAnimalTFData` for downloading and processing data.
Specify Arguments	Argument setup	Accept `filename` to determine the saving path for the downloaded file.
Download Data	File download	Use `urllib.request` to download data from the AnimalTFDB URL.
Read Data	Incremental reading	Read data in chunks of 1000 rows for memory efficiency.
Convert Data	Data conversion	Convert the processed list into a `pandas` DataFrame.
Save Data	File saving	Save the final DataFrame to a CSV file at the specified path.
Output Message	Confirmation	Output a success message upon completion of data extraction.

3.2.7 Integration of Transcription Factor Data with Signaling Pathways

The integration of transcription factor (TF) data from *AnimalTFDB* with signaling pathways identified through *Pathway Commons* and *CellTalk* is a critical step in constructing a comprehensive network of cellular signaling mechanisms in *Homo sapiens*. This process involves combining TFs, which are key regulatory components, with signaling networks (SN) to create an interconnected system of nodes (biological entities such as TFs, genes, and proteins) and edges (interactions or relationships between these entities). By integrating these datasets, we aim to systematically analyze the interactions between TFs and signaling pathways in humans, uncovering their roles in cellular processes and disease states, such as cancer and metabolic disorders.

Methodology for Combining TF and Signaling Networks in Humans

1. **Data Extraction and Preprocessing**:

- **Transcription Factor Data**: TFs specific to *Homo sapiens* were extracted from *AnimalTFDB*, a comprehensive database of TFs. Each TF was annotated with its associated target genes and regulatory functions in human cells.
- **Signaling Pathway Data**: Signaling pathways relevant to humans were retrieved from *Pathway Commons* and *CellTalk*, which provide curated information on protein-protein interactions, signaling cascades, and pathway cross-talk in human systems.
- **High-Throughput Data (HTD)**: Whole-genome sequencing and RNA sequencing data from the *Genomic Data Commons (GDC)*, specifically the TCGA (The Cancer Genome Atlas) dataset, were included to provide gene expression profiles and genomic alterations specific to human samples. These datasets were preprocessed to remove low-quality samples, correct technical biases, and normalize expression levels.

2. **Quality Control and Sample Filtering**:
   - **Rigorous Quality Control**: Quality control (QC) was performed to ensure the integrity of the datasets. Low-quality human samples were identified based on metrics such as read depth, mapping efficiency, and outlier expression profiles. Samples failing to meet predefined thresholds (e.g., low read counts, high technical noise, or poor alignment rates) were excluded from further analysis.
3. **Network Construction**:
   - TFs were mapped to their corresponding signaling pathways in humans based on their target genes and regulatory roles. This mapping was achieved by overlapping TF target genes with genes involved in specific signaling pathways in human cells.
   - The resulting network was represented as a graph, where nodes represent biological entities (e.g., TFs, genes, proteins) and edges represent interactions (e.g., regulatory relationships, protein-protein interactions) specific to human systems.
4. **Integration of High-Throughput Data**:
   - RNA sequencing data from human TCGA samples were used to validate the activity of TFs and signaling pathways by correlating gene expression levels with pathway activity scores.
   - Whole-genome sequencing data provided insights into genomic alterations (e.g., mutations, copy number variations) in human samples that could influence TF activity or signaling pathway dynamics.
5. **Functional Analysis**:
   - The integrated network was analyzed to identify key regulatory nodes and their functional implications in human disease states. For example, TFs that regulate multiple signaling pathways in humans were flagged as potential master regulators.
   - Pathway enrichment analysis was performed to identify signaling pathways significantly associated with specific disease conditions in humans, such as cancer or metabolic disorders.

Summary of Data Processing and Integration

The steps involved in data processing and integration are summarized in Table 3.6, which outlines each phase of data handling, from extraction and preprocessing to network construction and functional analysis, specifically for *Homo sapiens*. This multi-faceted approach ensures a robust and comprehensive analysis of TF-signaling pathway interactions in humans, providing insights into regulatory networks and potential therapeutic targets.

3.2.8 Database Structure Implementation

A well-structured MySQL database was established to manage and process data efficiently, facilitating subsequent analyses. The following steps describe the creation and organization of the database, the rationale behind its structure, and how it optimizes data storage and retrieval. Complete Python scripts used for database generation can be found in the supplementary materials in Table A.1, Appendix A.

Table 3.6: Data Processing and Integration Methods

Step	**Action**	**Details**
High-Throughput Data		
Preprocessing	Check for anomalies, biases, and technical issues	Analyze whole-genome and RNA sequencing data obtained from GDC.
Quality Control	Identify and remove low-quality samples	Ensure dataset integrity through rigorous quality assessment.
Filtering and Normalization	Filter out low-expression genes and normalize data	Adjust for technical variances to ensure comparability.
Differential Expression Analysis	Analyze gene expression differences	Conduct statistical comparisons across experimental conditions.
Signaling Pathway Data		
Pathway Extraction	Select relevant pathways based on the research question	Extract pathways from the Pathway Commons database relevant to the study focus.
Data Formatting	Prepare pathway data for integration	Ensure compatibility with high-throughput datasets for seamless analysis.

- **Database Schema Design:** A detailed schema was developed to accommodate heterogeneous data from various sources such as the Genomics Data Commons (GDC), AnimalTFDB 3.0, and Pathway Commons. The database schema was designed with normalization principles to ensure minimal data redundancy while maintaining high data integrity. The schema consists of multiple relational tables, each representing specific entities (e.g., genes, transcription factors, pathways). These tables are interconnected using foreign key relationships, ensuring consistency across datasets and allowing for efficient data integration.

- **Creation of the MySQL Database:** The database was implemented using MySQL, and the entire process—ranging from table creation to data population—was automated through Python scripts. For each entity in the schema, a corresponding table was generated in the database, with fields tailored to accommodate specific data types (e.g., VARCHAR, INT, FLOAT). The creation scripts also defined primary and foreign keys to enforce relational integrity. Data from external sources was imported into the database, ensuring that the structure supports a seamless integration of future datasets.

- **Data Organization:** Data from GDC, AnimalTFDB 3.0, and Pathway Commons were mapped onto specific tables according to their respective entities. For instance, gene-related information was stored in a designated table with fields capturing the gene identifiers, symbols, and associated metadata. Each dataset was organized in such a way as to preserve the original relationships within the data, while foreign keys linked related data across tables. This modular approach enhances the extensibility and maintainability of the database, allowing for additional datasets or entities to be incorporated without structural modifications.

- **Integration Rules:** Integration of data from diverse sources was handled by establishing uniform rules to ensure consistency. A key integration challenge was aligning identifiers across multiple datasets (e.g., gene identifiers). To address this, a standardized format for gene and transcription factor identifiers was adopted. Additionally, relational integrity was enforced using foreign keys, ensuring that each entry in a table is accurately linked to its corresponding entries in related tables. This integration strategy guarantees consistency and prevents data fragmentation across the database.

- **Indexing and Query Optimization:** To improve query performance, efficient indexing mechanisms were implemented. Indexes were created on frequently queried fields, such as gene identifiers and transcription factor names, significantly reducing query response time. In cases where queries involved multiple fields, compound indexes were employed to further enhance performance. Additionally, the database supports complex queries using SQL, allowing for retrieval of specific datasets based on multiple criteria. NoSQL techniques were also integrated where necessary to manage semi-structured data efficiently.

- **Database Structure and Optimization:** The overall structure of the database is optimized to ensure efficient data storage and retrieval. By adhering to the principles of third normal form (3NF), the schema minimizes redundancy, reducing storage overhead while maintaining data integrity. The use of foreign keys ensures that relationships between datasets are preserved, enabling accurate and efficient data joins. Furthermore, the indexing and query optimization strategies employed ensure that even large datasets can be queried with minimal delay, thereby enhancing the overall performance of the system.
- **Rationale for Structural Design:** The design of the database structure ensures that data is stored efficiently, with minimal redundancy and optimal organization for retrieval. The normalization of tables reduces duplication of data, while the use of primary and foreign keys maintains the integrity of relationships between different datasets. This design is particularly suited to handle the complexity and size of biological datasets, ensuring that both current and future data additions can be managed effectively without necessitating substantial structural changes.

3.2.9 Data Processing and Integration

The integration of mutation data from the *The Cancer Genome Atlas (TCGA)* dataset into the pre-existing network, constructed from transcription factor (TF) and signaling pathway data, involved the following steps:

- **Mutation Data Extraction:**
  - Somatic mutation data for *Homo sapiens* were extracted from the TCGA dataset, focusing on specific genes of interest that are relevant to the signaling pathways and transcription factors in the pre-existing network.
  - Only non-synonymous mutations (e.g., missense, nonsense, and frameshift mutations) were retained, as these are more likely to have functional consequences on protein activity and signaling pathways.
- **Quality Control and Filtering:**
  - Low-quality mutation calls were filtered out based on TCGA quality control metrics, such as read depth (minimum threshold of 10x coverage) and variant allele frequency (minimum threshold of 5%).
  - Mutations with ambiguous or incomplete annotations were excluded to ensure data reliability.
- **Integration with Pre-Existing Network:**
  - The filtered mutation data were mapped to the corresponding genes in the pre-existing network using unique gene identifiers (e.g., Ensembl IDs or gene symbols).
  - Mutations were added as node attributes to the network, allowing for the identification of genes with altered function due to somatic mutations.
  - The updated network was visualized and analyzed using *Cytoscape* to explore the impact of mutations on signaling pathways and transcription factor activity.

Summary of Data Processing and Integration

The steps involved in integrating mutation data into the pre-existing network are summarized in Table 3.6, which outlines the key phases of data extraction, quality control, and network integration. This approach enabled the identification of genes with functional mutations that may influence signaling pathways and regulatory networks in human disease states.

3.2.10 Data Filtering and Creation of Initial Network

The construction of the initial network involved a systematic process of data extraction, filtering, and network representation. This subsection describes the methodology used to filter the data and create the initial network.

Data Extraction from Pathway Commons

The Pathway Commons dataset was used as the primary source of interaction data due to its comprehensive coverage of biological pathways and interactions. The following steps were taken to extract and filter the data:

Downloading the Dataset - The entire Pathway Commons dataset was downloaded in a standardized format (e.g., SIF or BioPAX) from the official Pathway Commons website (https://www.pathwaycommons.org/). - The dataset includes interactions from multiple sources, such as KEGG, Reactome, and PubMed, ensuring broad coverage of biological pathways.

Loading the Dataset into Python The dataset was loaded into Python using the ‘pandas‘ library for data manipulation and analysis. The following code snippet demonstrates how the dataset was loaded:

```
import pandas as pd

# Load the Pathway Commons dataset
pathway_commons_data = pd.read_csv('pathway_commons_data.tsv', sep='\t', comment='#')
```

Inspecting the Dataset The dataset was inspected to understand its structure and contents. Key columns include: - **PARTICIPANT_A**: The first interacting entity (e.g., gene, protein). - **PARTICIPANT_B**: The second interacting entity. - **INTERACTION_TYPE**: The type of interaction (e.g., physical interaction, transcriptional regulation). - **PATHWAY_NAMES**: The biological pathways associated with the interaction. - **PUBMED_IDS**: PubMed references supporting the interaction.

Filtering the Dataset

To ensure relevance and quality, the dataset was filtered based on specific criteria, including KEGG pathways and PubMed IDs.

Filtering Based on KEGG Pathways Interactions were filtered to include only those associated with KEGG pathways:

```
# Filter interactions based on KEGG pathways
kegg_filtered_data = pathway_commons_data[
    pathway_commons_data['PATHWAY_NAMES'].str.contains('KEGG', na=False)
]
```

Filtering Based on PubMed IDs Interactions were further filtered to include only those supported by PubMed references:

```
# Filter interactions based on PubMed IDs
pubmed_filtered_data = kegg_filtered_data[
    kegg_filtered_data['PUBMED_IDS'].notna()
]
```

Saving the Filtered Dataset The filtered dataset was saved for further processing:

```
# Save the filtered dataset
pubmed_filtered_data.to_csv('filtered_pathway_commons_data.tsv', sep='\t', index=False)
```

Creating the Initial Network

The filtered dataset was used to construct the initial network.

Network Representation The filtered interactions were represented as a graph using the ‘NetworkX‘ library:

```
import networkx as nx

# Create a directed graph
```

```
G = nx.DiGraph()

# Add edges to the graph
for _, row in pubmed_filtered_data.iterrows():
    G.add_edge(row['PARTICIPANT_A'], row['PARTICIPANT_B'], interaction_type=row['INTERACTION_TYPE'])
```

Visualizing the Network The initial network was visualized using ‘matplotlib‘:

```
import matplotlib.pyplot as plt

# Visualize the network
pos = nx.spring_layout(G)
nx.draw(G, pos, with_labels=True, node_size=50, font_size=8, edge_color='gray')
plt.title('Initial Network')
plt.show()
```

Storing the Network The network was saved in a structured format for further analysis:

```
# Save the network
nx.write_graphml(G, 'initial_network.graphml')
```

3.2.11 Network Modeling

Network modeling integrates data from multiple databases to construct, refine, and validate network models. The process involves the following steps:

- **Initial Network Construction:**
  - A foundational network was constructed using signaling pathway data from *Pathway Commons* and transcription factor (TF) data from *AnimalTFDB*. Nodes in the network represent biological entities (e.g., genes, proteins, TFs), and edges represent interactions (e.g., regulatory relationships, protein-protein interactions).
  - The network was enriched with additional interactions from curated databases such as *CellTalk* and *PathwayCommon* to ensure comprehensive coverage of known signaling pathways and regulatory mechanisms.
- **Integration of High-Throughput Data:**
  - Processed mutation data from the *The Cancer Genome Atlas (TCGA)* dataset were integrated into the network. These data were mapped to corresponding genes in the network to identify genes with functional alterations.
  - The integration focused on adding mutation information as node attributes, enabling the identification of genes with somatic mutations that may influence signaling pathways and regulatory mechanisms.
- **Network Validation and Refinement:**
  - The constructed network was validated by comparing it to well-established models of key signaling pathways (e.g., p53, *NF-κB*, ATM). This comparison was not intended to achieve exact agreement but to ensure that the network captures known interactions and regulatory mechanisms.
  - Discrepancies between the constructed network and established models were analyzed to identify potential gaps or novel interactions. For example:
    - If the constructed network contained additional nodes or edges not present in established models, these were evaluated for biological relevance using literature and functional annotations.
    - If the constructed network lacked nodes or edges present in established models, the data sources and integration methods were revisited to ensure no critical interactions were missed.

- The network was iteratively refined based on validation results, ensuring robustness and biological plausibility. Novel interactions identified during this process were flagged for further experimental validation.

Summary of Network Modeling

The network modeling process is summarized in Table 3.7, which outlines the key steps, methods, and outcomes of each phase. This approach ensures that the network is both comprehensive and biologically accurate, while also allowing for the discovery of novel interactions that may not be present in existing models.

Table 3.7: Summary of Network Modeling Steps

Step	Methods and Data Sources	Outcomes
Initial Network Construction	• Signaling pathway data from *Pathway Commons*. • Transcription factor data from *AnimalTFDB*. • Additional interactions from *CellTalk* and *PathwayCommon*.	• A foundational network with nodes (genes, proteins, TFs) and edges (interactions). • Comprehensive coverage of known signaling pathways and regulatory mechanisms.
Integration of High-Throughput Data	• Processed mutation data from TCGA. • Mapping of mutations to corresponding genes in the network.	• Identification of genes with functional alterations. • Mutation information added as node attributes.
Network Validation and Refinement	• Comparison to established models (e.g., p53, *NF-κB*, ATM). • Analysis of discrepancies to identify gaps or novel interactions. • Iterative refinement based on validation results.	• Confirmation of known interactions and regulatory mechanisms. • Identification of novel interactions for further experimental validation. • A robust and biologically plausible network.

3.3 Signal Flow Modeling

Signal Flow Modeling (SFM) studies how information flows from input components (e.g., receptors) to output elements (e.g., transcription factors). The flux of signal propagation defines the path from the source to the sink node. Methods include:

3.3.1 Signal Flow Modeling Methods

Signal flow modeling approaches simulate and study the flow of information across biological networks. The Python code generated for each model is in appendices A. This study used the following methodologies to model signal propagation:

- **Random Walk Model**: Simulates diffusion-like processes across the network, with each step representing a node change impacted by probability based on network structure or edge weights.
- **PageRank Algorithm**: PageRank, which was originally intended for web search engines, quantifies the relevance of nodes in a network using the concept of "voting" or "recommendation," indicating the influence of one node over another via links.
- **Boolean Model**: Signaling events are represented as binary states (on/off), which simplifies complicated regulatory networks into logical connections for the study of regulatory dynamics and stable states.
- **RCNN Model**: The Recurrent Convolutional Neural Network (RCNN) combines recurrent connections with convolutional layers to capture temporal relationships and spatial patterns in sequential data. It is useful for dynamic investigation of signal propagation in biological networks.

These approaches were chosen for their capacity to capture various elements of signal flow within biological networks, allowing thorough research of signaling dynamics and network behavior.

These strategies will be rigorously tested to determine their efficacy across various network sizes. The most promising method will be chosen for more research and analysis.

3.3.2 Network Visualization and Analysis

Integrated high-throughput sequencing and signaling pathway data were visualized and analyzed using Cytoscape, a powerful tool for exploring relationships, patterns, and biological processes within networks. The following steps outline the process:

Installation and Setup

Cytoscape was downloaded and installed from the official website[1]. The software was configured to meet the analytical requirements of this study.

Importing Data

Gene interaction data were imported into Cytoscape using compatible formats such as CSV, SIF, or XGMML. The steps for data import were as follows:

1. Accessed the **File** menu in Cytoscape.
2. Selected the **Import** option and chose either **Network** or **Table** based on the data type.
3. Navigated to the file location and designated it for import.
4. Followed on-screen instructions to specify import preferences and configure data loading.

Network Layout

After importing the data, the gene interaction network was displayed, with genes as nodes and interactions as edges. The network was organized using the following steps:

1. Explored various layout algorithms, such as force-directed, circular, and hierarchical layouts.
2. Selected the most appropriate layout to create a clear and interpretable visualization.

Styling and Visual Enhancements

The visual representation of the network was enhanced using Cytoscape's styling options:

1. Applied visual styles to nodes and edges to differentiate genes or highlight specific attributes.
2. Customized node size, color, shape, and labels based on gene properties.
3. Adjusted edge thickness, color, and style to reflect interaction strength or type.
4. Utilized features like node clustering, grouping, and subnetwork highlighting for improved interpretation.

Advanced Network Analysis

Cytoscape's advanced analysis tools were used to gain deeper insights into the gene interaction network. The following analyses were performed:

- **Centrality Analysis**:
  - **Betweenness Centrality**: Identified genes that act as bridges between other genes, highlighting critical intermediaries in the network.
  - **Eigenvector Centrality**: Identified highly connected genes that may serve as key regulatory hubs.

[1] https://cytoscape.org/

- **Clustering Analysis**: Detected tightly coupled gene clusters or modules using algorithms such as hierarchical clustering and k-means clustering.
- **Modular Analysis**: Identified distinct communities of genes with shared functional characteristics, revealing insights into biological processes and signaling pathways.

Exporting and Saving

The final network visualization and analysis results were saved for future reference:

1. Saved the session as a Cytoscape session file (.cys) to retain network data, layout, and visual styles.
2. Exported the network visualization as an image file (e.g., PNG, JPEG, or PDF) or a vector graphics file (e.g., SVG or EPS) for inclusion in publications or presentations.

3.3.3 Runtime Analysis

To evaluate the efficiency of the network visualization process, runtime analysis was performed:

- Measured the time required for network visualization across the dataset and network scales.
- Recorded and analyzed runtime under various settings to identify potential optimizations.

3.3.4 Network Transformation from Cytoscape to Python

After visualization and analysis in Cytoscape, the network was converted into a Python-compatible format to facilitate further computational analysis using the `NetworkX` framework [95]. Initially, node and edge interactions were extracted from the database containing all relevant datasets. Key interactions were identified, organized, and stored in a CSV file using Python. The network was first visualized in Cytoscape for preliminary exploration and then transferred to Python for more in-depth analysis. This conversion enabled the application of advanced algorithms and comprehensive computational techniques, significantly enhancing the understanding of the network's structural properties and its evolving behavior.

3.4 PageRank Algorithm for Network Analysis

3.4.1 Introduction to PageRank

The PageRank algorithm, initially developed by Larry Page and Sergey Brin at Stanford University, is a fundamental method for ranking nodes based on their importance in a network [96]. Though first applied to rank web pages, its application has expanded to network analysis across various domains, including biological networks. In such systems, PageRank identifies influential nodes, such as key proteins or genes in interaction networks, by evaluating both the quantity and quality of their connections.

3.4.2 Algorithmic Implementation in Biological Networks

In this study, the PageRank algorithm was utilized to evaluate the centrality and importance of nodes in a biological signaling network. The `NetworkX` Python library, a powerful tool for network analysis, was employed to compute the PageRank scores for each node. This approach provides insights into the relative significance of individual components (e.g., proteins, genes, or other biomolecules) within the network structure. By applying this algorithm, key nodes that play critical roles in information flow, signaling pathways, or network stability can be identified, offering a deeper understanding of the network's functional organization and potential regulatory mechanisms. This method is broadly applicable to various biological networks, enabling researchers to prioritize nodes for further experimental validation.

Basic PageRank Formula

The PageRank algorithm assigns a rank to each node in a network based on the ranks of the nodes that link to it, creating a recursive ranking system. The PageRank score $PR(i)$ of a node i is calculated using the following formula:

$$PR(i) = \frac{1-d}{N} + d \sum_{j \in M(i)} \frac{PR(j)}{L(j)}, \tag{3.1}$$

where:

- $PR(i)$ is the PageRank score of node i,
- d is the damping factor, typically set to 0.85, which represents the probability that a random walker will continue exploring the network,
- N is the total number of nodes in the network,
- $M(i)$ is the set of nodes that link to node i,
- $L(j)$ is the number of outbound links from node j.

The damping factor d ensures that the algorithm converges and accounts for the likelihood of a random jump to any node in the network. This formula is widely used to identify influential nodes in various types of networks, including biological, social, and information networks [97].

This implementation can be adapted to analyze biological networks, such as protein-protein interaction networks or signaling pathways

This formula ensures that nodes receive a higher PageRank score if they are linked by other important nodes, reflecting their centrality in the network. For biological networks, this is essential to identifying nodes that influence multiple pathways.

3.4.3 Clarifying the Role of PageRank in Network Analysis

The use of PageRank in biological networks fulfills three important roles:

- **Ranking Method:** PageRank ranks nodes based on their importance in the network. In biological networks, this ranking helps to identify critical regulatory genes or proteins [97].
- **Centrality Measure:** As a centrality measure, PageRank takes into account not only the number of connections but also the importance of the nodes that are connected. This helps to identify nodes that play pivotal roles as hubs in the network [98].
- **Network Structure Analysis:** PageRank also helps to uncover the hierarchical structure of the network by identifying the most influential nodes. This can reveal important pathways or potential points of failure in biological systems [99].

In this way, PageRank functions as both a local and global centrality measure, providing a comprehensive perspective on the structure of biological networks.

3.4.4 Comparison with Other Network Metrics

In addition to PageRank, three other centrality measures were applied to the network to provide a more complete analysis. These measures are Degree Centrality, Eigenvector Centrality, and Betweenness Centrality.

Degree Centrality

Degree Centrality is one of the simplest metrics and measures the number of direct connections a node has. The Degree Centrality $DC(i)$ of a node i is defined as:

$$DC(i) = \frac{\deg(i)}{N-1}, \tag{3.2}$$

where $\deg(i)$ is the number of edges connected to node i, and N is the total number of nodes in the network. Although Degree Centrality provides insight into the local importance of a node, it does not account for the quality or significance of these connections [100].

Eigenvector Centrality

Eigenvector Centrality improves upon Degree Centrality by not only considering the number of connections but also the importance of the nodes to which a node is connected. The Eigenvector Centrality $EC(i)$ of a node i is defined as:

$$EC(i) = \frac{1}{\lambda} \sum_{j \in M(i)} A_{ij} \cdot EC(j), \tag{3.3}$$

where:

- A_{ij} is the adjacency matrix of the network, indicating whether there is a direct link between nodes i and j,
- $EC(j)$ is the Eigenvector Centrality of node j,
- λ is the largest eigenvalue of the adjacency matrix.

Eigenvector Centrality is valuable for identifying influential nodes, especially those that are connected to other highly influential nodes [101].

Betweenness Centrality

Betweenness Centrality measures the extent to which a node lies on the shortest path between other nodes. The Betweenness Centrality $BC(i)$ of a node i is given by:

$$BC(i) = \sum_{s \neq i \neq t} \frac{\sigma_{st}(i)}{\sigma_{st}}, \tag{3.4}$$

where:

- σ_{st} is the total number of shortest paths between nodes s and t,
- $\sigma_{st}(i)$ is the number of those paths that pass through node i.

Betweenness Centrality is useful for identifying nodes that serve as bridges or bottlenecks in the network, as these nodes control the flow of information between other nodes [102].

3.4.5 Results Interpretation and Issues Addressed

In analyzing a biological signaling network, the PageRank algorithm identified key nodes (e.g., proteins or genes) as central to the network. Several specific considerations were made in interpreting these results:

- **Elevated PageRank Values:** Nodes with significantly higher PageRank values were categorized as having "elevated" ranks. To define this, the mean μ and standard deviation σ of the PageRank scores were computed, and nodes with $PR(i) > \mu + \sigma$ were considered elevated.
- **Comparative Node Analysis:** Differences in PageRank between nodes were analyzed in terms of biological significance. For example, a node with a lower PageRank score might still have greater biological importance due to its role in critical pathways or processes.
- **Probability Distribution of PageRank Values:** The PageRank scores were normalized to form a probability distribution, which facilitated a probabilistic interpretation of node importance. This helps in understanding the hierarchy and influence distribution within the network.

3.4.6 Convergence and Runtime Considerations

The PageRank algorithm iteratively refines the ranking of nodes until convergence, which is achieved when the change in PageRank scores between iterations falls below a specified threshold. For the analyzed network, the algorithm converged in a reasonable number of iterations due to the manageable size of the network. The time complexity of the PageRank algorithm is $O(N + E)$, where N is the number of nodes and E is the number of edges, making it computationally efficient even for larger biological networks [97].

The PageRank algorithm proved to be a versatile and robust tool for network analysis in this study. It successfully identified critical nodes within the biological signaling network by evaluating node centrality and importance. When compared with other centrality measures such as Degree Centrality, Eigenvector Centrality, and Betweenness Centrality, PageRank offered a more balanced view of the network's structure, considering both direct and indirect connections. This comprehensive analysis revealed key insights into the regulatory mechanisms within the network [103].

3.5 Random Walk Algorithm

3.5.1 Overview

The Random Walk Algorithm is a stochastic technique frequently used in network analysis, particularly within biological networks. Unlike deterministic approaches such as PageRank, this algorithm simulates the movement of an entity, or "walker," across a network of nodes (e.g., genes or proteins) connected by edges representing biological interactions. The algorithm probabilistically traverses the network based on edge weights, which encode interaction strengths, confidence levels, or biological significance, thus uncovering latent patterns, determining node importance, and revealing the structural organization of complex biological systems [104].

In this work, the random walk algorithm's edge weights are derived from mutation datasets, which are utilized to model mutation-driven biological interactions. Initially, each node (representing a gene or protein) is assigned a uniform weight of 1, reflecting equal baseline significance derived from mutation data. The weights are then probabilistically distributed across the network, with each edge's weight reflecting the interaction strength between nodes as determined by their biological relevance. This weighted probabilistic approach enables a dynamic flow of information through the network, facilitating the identification of key patterns, regulatory elements, and functional modules.

3.5.2 Mathematical Formulation and Procedure

The Random Walk Algorithm is a stochastic process that traverses a network by moving from one node to another based on predefined probabilities [105]. The algorithm can be mathematically described through the following steps:

1. **Initialization**: The random walk begins by selecting an initial node i from a probability distribution P_0. Here, P_0 represents the initial probability distribution over all nodes in the network, which determines the likelihood of starting the walk at a particular node. This distribution can be:

    - **Uniform**: If no prior information is available, P_0 is uniform, meaning each node has an equal probability of being selected as the starting point:

    $$P_0(i) = \frac{1}{N}, \quad \forall i \in \text{Nodes},$$

    where N is the total number of nodes in the network [106].

    - **Biased**: If prior biological knowledge or data (e.g., mutation data) is available, P_0 can be biased to reflect the importance or relevance of specific nodes. In this work, the initial weights of nodes are derived from mutation data, setting each node's weight to 1:

    $$w_i = 1, \quad \forall i \in \text{Nodes}.$$

    The initial probability distribution P_0 is then computed as:

    $$P_0(i) = \frac{w_i}{\sum_{k \in \text{Nodes}} w_k}, \quad \forall i \in \text{Nodes}.$$

    This approach is commonly used in biological network analysis [107].

2. **Transition Probability Calculation**: Once the walker is at a node i, it moves to a neighboring node j with a probability proportional to the edge weight w_{ij}. The edge weights reflect the biological significance or strength of interactions between nodes, as derived from mutation data. The transition probability from node i to node j is given by:

$$P(i \to j) = \frac{w_{ij}}{\sum_{k \in \mathcal{N}(i)} w_{ik}},$$

where:

- $\mathcal{N}(i)$ is the set of neighboring nodes of node i,
- w_{ij} is the edge weight between nodes i and j,
- $\sum_{k\in\mathcal{N}(i)} w_{ik}$ is the sum of edge weights for all neighbors of node i.

This formulation ensures that the walker is more likely to traverse edges with higher weights, reflecting stronger biological interactions [108].

3. **Random Walk Execution**: The walker iteratively transitions between nodes according to the computed transition probabilities. Let S be the number of steps taken. At each step t, the walker moves as follows:

$$i_{t+1} = \begin{cases} j, & \text{with probability } P(i_t \to j), \\ i_t, & \text{with probability } 1 - \sum_{j\in\mathcal{N}(i_t)} P(i_t \to j), \end{cases}$$

where:

- i_t is the current node at step t,
- j is a neighboring node of i_t,
- $P(i_t \to j)$ is the transition probability from i_t to j.

This process is repeated for a specified number of steps L. The walk may terminate earlier if a convergence criterion is met, such as when the node visitation probabilities stabilize [97].

4. **Data Collection**: During the random walk, data is collected on:

- Node visitation frequencies f_i, which count how often each node i is visited,
- Sequences of node visits, which record the order in which nodes are traversed,
- Edge traversal patterns, which track how often each edge is traversed.

This data is used to generate a visitation distribution $P(v)$, where v denotes a node. The visitation frequency for node i is computed as:

$$f_i = \text{Number of visits to node } i.$$

This approach is widely used in network analysis to identify important nodes [98].

5. **Analysis and Interpretation**: The visitation distribution and traversal patterns are analyzed to uncover key properties of the network, such as:

- **Node significance**: Nodes with higher visitation frequencies are considered more important or central in the network [109].
- **Community structures**: Groups of nodes that are frequently visited together may indicate functional modules or communities [110].
- **Network topology**: The overall structure of the network, including connectivity and robustness, can be inferred [111].

The visitation frequencies can be normalized to obtain a probability distribution:

$$\hat{f}_i = \frac{f_i}{\sum_j f_j},$$

where $\hat{f}_i$ represents the normalized visitation frequency for node i. Additional metrics, such as centrality measures (e.g., betweenness, closeness) and clustering coefficients, can be computed to further interpret the biological relevance of nodes and substructures [112].

3.5.3 Uncovering Latent Patterns

The Random Walk Algorithm reveals latent patterns in biological networks by examining visitation distributions and node transition sequences. Repeated random walks highlight clusters of nodes that are frequently co-visited, indicating potential communities or functional modules. The clustering coefficient C_i for node i can be defined as:

$$C_i = \frac{2E_i}{k_i(k_i - 1)}, \tag{3.5}$$

where E_i is the number of edges between the k_i neighbors of node i. This formula is a standard measure of local clustering in networks [113]. Aggregating the walk data allows for identifying regions in the network where nodes are strongly interconnected, reflecting biological modularity or shared functions. The transition probability matrix, constructed during the random walk, helps detect recurring pathways and interaction patterns that may not be evident from static network analysis.

3.5.4 Determining Node Significance

Node significance is derived from the frequency of node visitation during the random walks. Nodes attracting a higher fraction of the total visits are considered more central or influential, suggesting biological importance, such as being key regulators or network hubs. The significance score for a node i is computed as:

$$S_i = \frac{f_i}{\sum_j f_j}, \tag{3.6}$$

where:

- f_i is the frequency of visits to node i,
- $\sum_j f_j$ is the sum of visitation frequencies for all nodes in the network.

The significance score S_i reflects the relative importance of node i in the network. A higher score indicates that the node is visited more frequently during the random walks, implying a more central or influential role in the network's information flow [114].

3.5.5 Node Significance Ranking

Let $G = (V, E)$ be a network with node set V and edge set E. The significance ranking $\mathcal{R} : V \to \mathbb{N}^+$ is constructed from significance scores $\{S_i\}_{i \in V}$ as follows:

For each node $i \in V$, its significance score S_i is computed as:

$$S_i = \frac{f_i}{\sum_{j \in V} f_j}$$

where f_i is the visitation frequency of node i during random walks on G.

Ranking Rules

The ranking $\mathcal{R}$ is determined by:

1. **Strict ordering**: For any two nodes $i, j \in V$,

$$\mathcal{R}(i) < \mathcal{R}(j) \iff S_i > S_j$$

2. **Tie handling**: If $S_i = S_j$, then $\mathcal{R}(i) = \mathcal{R}(j)$
3. **Rank assignment**: The node with highest S_i receives $\mathcal{R}(i) = 1$, the next highest receives 2, and so forth

This ranking scheme guarantees:

- **Monotonicity**: Higher scores always correspond to better ranks (lower rank numbers)
- **Completeness**: All nodes are assigned a rank (with possible ties for equal scores)
- **Interpretability**: Rank 1 identifies the most significant node

The edge weights w_{ij} influence these rankings through their effect on the random walk transition probabilities, making nodes connected by biologically significant edges more likely to achieve higher scores and consequently better ranks.

3.5.6 Elucidating the Structure of Complex Networks

The Random Walk Algorithm helps elucidate the structure of complex biological networks by uncovering topological features such as hubs, bottlenecks, and communities. The degree centrality D_i for a node i is defined as:

$$D_i = \sum_j w_{ij}, \tag{3.7}$$

where w_{ij} is the weight of the edge between nodes i and j. By analyzing node transition sequences and their associated probabilities, the algorithm reveals hierarchical or modular structures. Clusters of nodes with high intra-group connectivity and low inter-group connectivity are detected as functional modules or biological communities [115]. The algorithm also helps identify weakly connected components or critical bridges, providing insights into the network's resilience and connectivity.

Furthermore, the temporal analysis of random walk sequences simulates biological signal diffusion, allowing the tracing of potential signaling pathways, protein interaction cascades, or gene regulatory networks. The temporal weight $w_{ij}(t)$ can be modeled to reflect changes over time:

$$w_{ij}(t) = w_{ij} \cdot \exp(-\lambda t), \tag{3.8}$$

where λ is a decay parameter. Temporal dynamics can be incorporated by varying edge weights over time or introducing time-dependent biases in the random walk, allowing a more nuanced exploration of pathways influenced by temporal factors, such as gene expression changes during different biological conditions [116].

3.5.7 Temporal Analysis of Random Walks

This section introduces a crucial distinction in my temporal analysis approach. While traditional temporal analyses in biological networks often focus on simulating molecular concentration changes over time, our method instead considers the **time-varying strength of interactions** between components. Specifically:

- **Static pathways**: Represent fixed interaction topologies (who-activates-whom) typically analyzed at steady-state
- **Dynamic molecular concentrations**: Track changes in molecule quantities over time (common in signaling pathway studies)
- **temporal approach**: Models *evolving interaction strengths* ($w_{ij}(t)$ while keeping node identities constant

This framework treats the network as a non-stationary system where edge weights may change to reflect:

- Varying regulatory strengths under different conditions
- Time-dependent interaction probabilities
- Context-specific pathway preferences

The time-dependent transition probability is defined as:

$$P(i \to j, t) = \frac{w_{ij}(t)}{\sum\limits_{k \in \mathcal{N}(i)} w_{ik}(t)}, \tag{3.9}$$

where:

- $w_{ij}(t)$: Interaction strength between nodes i and j at time t
- $\mathcal{N}(i)$: Neighborhood of node i
- The denominator ensures proper probability normalization

3.5.8 Context-Aware Biasing

To prioritize biologically relevant pathways during specific conditions, I introduce a temporal biasing mechanism:

$$P_{\text{biased}}(i \to j, t) = \alpha P(i \to j, t) + (1 - \alpha) \cdot B_j(t), \tag{3.10}$$

Key components:

- $B_j(t)$: Time-dependent node importance (e.g., experimental measurements)
- $\alpha \in [0, 1]$: Balances topological vs. biological relevance
- Adapts to condition-specific network configurations

This approach captures essential features of:

- Pathway switching between cellular states
- Condition-dependent regulatory mechanisms
- Dynamic response to external stimuli

The model's behavior adapts to temporal changes in edge weights while maintaining constant node composition, reflecting how biological systems rewire their interaction strengths under different conditions.

3.5.9 Ergodicity in Random Walks

Ergodicity is a key property in the context of the Random Walk Algorithm. A stochastic process is considered ergodic if its time averages converge to ensemble averages over the long run. In biological network analysis, this means that given enough steps, the random walker will visit all nodes in the network with a probability reflecting their relative significance, regardless of the starting node.

For a network to exhibit ergodic behavior, it must be strongly connected, meaning there is a path from any node to any other node. In the context of biological networks, this implies that even sparsely connected components can eventually influence the overall structure as the random walker explores the network. The necessary condition for ergodicity can be mathematically formulated as:

$$\lim_{t \to \infty} P(i \to j, t) = \pi_j, \tag{3.11}$$

where π_j is the stationary distribution of node j indicating its long-term visitation probability. This property ensures that insights gleaned from random walks are comprehensive and not unduly biased by initial conditions [114].

Monitoring the convergence of visitation frequencies during the random walk allows researchers to confirm the ergodic nature of the network, ensuring that the resultant analysis is robust and representative of the underlying biological interactions.

3.5.10 Applications in Network Analysis

The Random Walk Algorithm is applied to several network analysis tasks in this work. While the mathematical formulation of the algorithm has been discussed earlier, the primary metric used in this study is the **visitation counter**, which tracks the frequency of visits to each node during the random walks. This metric serves as the foundation for deriving insights into network structure and function. Below, we outline the key applications of the algorithm:

- **Community Detection**:
  Co-visitation patterns and edge traversal frequencies help identify communities within the biological network. Nodes that are frequently visited together are likely part of the same functional module or biological pathway. The visitation counter f_i for each node i is used to group nodes into communities based on their co-occurrence in random walk trajectories. This approach leverages the ergodic property of random walks, ensuring that the visitation frequencies reflect the long-term behavior of the walker and provide a robust representation of network communities [115].

- **Determining Key Nodes (Significance)**:
  Nodes visited more frequently during random walks are considered more significant in the network. This aligns with centrality measures like betweenness and degree centrality but incorporates a probabilistic approach that captures the dynamic flow of biological information. The significance score S_i for node i is computed as:

  $$S_i = \frac{f_i}{\sum_j f_j},$$

  where:

  - f_i is the visitation frequency of node i,
  - $\sum_j f_j$ is the total number of visits across all nodes.

  Nodes with higher S_i values are ranked higher in significance, indicating their central or influential role in the network. This ranking is robust due to the ergodic nature of the random walk, which ensures that the significance scores are representative of the network's long-term behavior [117].

- **Identifying Signaling Pathways and Information Flow**:
  The sequence of node visits simulates the propagation of biological signals. Analyzing these temporal visitation patterns helps identify key signaling pathways and potential routes for information diffusion. The visitation counter f_i and the sequence of visits provide insights into the flow of information through the network. By leveraging the ergodic property, the algorithm ensures that the identified pathways are not biased by the starting node and reflect the true structure of the network [116].

- **Biased Exploration of Specific Regions**:
  Biases can be introduced to prioritize certain nodes or pathways, allowing targeted exploration of specific substructures relevant to particular biological processes or diseases. The bias is incorporated into the transition probability as follows:

  $$P(i \to j) = \alpha \cdot \frac{w_{ij}}{\sum_{k \in \mathcal{N}(i)} w_{ik}} + (1 - \alpha) \cdot B_j,$$

  where:

  - α is a tuning parameter ($0 \leq \alpha \leq 1$) that controls the balance between the original transition probability and the bias,
  - B_j is the bias factor for node j, which reflects its biological relevance or importance in the context of the study.

  By adjusting α and B_j, the algorithm can be directed to explore specific regions of the network more intensively, providing deeper insights into targeted biological processes. The ergodic property ensures that even with biased exploration, the algorithm converges to a meaningful representation of the network over time [116].

3.5.11 Algorithmic Parameters and Performance Considerations

Several parameters influence the effectiveness of the Random Walk Algorithm in uncovering patterns and determining node significance:

- **Number of Walkers and Steps**: Multiple walkers (N_w) perform random walks, each taking a fixed number of steps (L). Increasing N_w and L improves network coverage and analysis precision [114].
- **Edge Weight Assignment**: Edge weights are based on factors like interaction strength or experimental confidence. Here, weights are derived from mutation datasets, with each node initially assigned a weight of 1. Weights are normalized to ensure that transition probabilities sum to one at each step [115].
- **Bias Introduction**: Biases can be introduced to prioritize nodes with higher degrees or biological relevance, guiding more focused exploration [116].
- **Convergence**: Convergence is monitored by analyzing the stabilization of visitation frequencies. Ergodicity ensures complete network exploration with a sufficiently large number of steps [114].

The computational complexity of the Random Walk Algorithm is influenced by the number of walkers (N_w), the number of steps (L), and the network size. The complexity scales linearly:

$$O(N_w \cdot L),$$

, making it suitable for large biological networks. Parallelization techniques can further accelerate computations by executing multiple walks simultaneously [116].

3.5.12 Automated Data Collection and Network Construction

The construction of the network was performed using data from multiple biological databases, including the Genomic Data Commons (GDC), Pathway Commons, CellTalk, and the AnimalTF Database. By sourcing data from these repositories, we ensured that the information was derived from experimental evidence rather than solely relying on literature reviews. These databases provided comprehensive information on molecular interactions, including activation and inhibition relationships, as well as edge directions for creating a directed graph representation of the signaling pathway.

The collected data was processed to transform raw interaction information into a directed graph, following these steps:

1. **Extraction of Interaction Pairs**: Extract interaction pairs (nodes and edges) from each database, ensuring consistency in naming conventions and interaction types. The datasets utilized in this step include sources like GDC for gene expression data and Pathway Commons for pathway interactions and regulatory relationships.

2. **Direction Assignment**: Assign directions to the edges based on the nature of the interactions (e.g., activation or inhibition), as indicated by the annotations in the respective databases. This process ensures that the interactions are accurately represented, reflecting biological reality.

3. **Conflict Resolution**: Resolve conflicting interaction information by prioritizing high-confidence interactions, as indicated by the evidence levels provided by the databases. This step is crucial for ensuring the reliability and accuracy of the constructed network.

4. **Edge Weight Assignment from Mutation Data**: Edge weights were assigned probabilistically based on mutation data from The Cancer Genome Atlas (TCGA). The mutation data provides information on the frequency and functional impact of mutations in genes, which can be used to infer the strength of interactions between nodes in the network. The process of assigning edge weights is as follows:

3.5.13 Initial Weight Assignment and Biological Relevance

- **Initial Weight Assignment**: At the start, all edges in the network are assigned a uniform weight of 1, i.e., $w_{ij} = 1$ for all edges (i, j). This represents an unbiased baseline where all interactions are considered equally likely [114].

- **Biological Relevance Measurement**: Biological relevance is quantified using mutation data from TCGA. For each gene (node) in the network, the mutation probability p_i is calculated as the frequency of mutations observed in the TCGA dataset. This probability reflects the likelihood that the gene is mutated in the studied biological context. The mutation probability p_i is computed as:

  $$p_i = \frac{\text{Number of samples with mutations in gene } i}{\text{Total number of samples}}.$$

  This formula is derived from standard statistical methods for calculating mutation frequencies [118].

- **Transforming Relevance into Edge Weights**: The interaction strength between two nodes i and j is derived from the mutation probabilities of the corresponding genes. If the interaction between i and j is supported by experimental evidence (e.g., from Pathway Commons or CellTalk), the edge weight w_{ij} is updated as:

  $$w_{ij} = p_i \cdot p_j \cdot c_{ij},$$

  where:

  - p_i and p_j are the mutation probabilities of nodes i and j, respectively,

- c_{ij} is a confidence score for the interaction between i and j, derived from the database annotations (e.g., experimental evidence level). The confidence score c_{ij} is normalized to the range [0, 1], where 1 indicates the highest confidence [119].

- **Normalization**: The edge weights are normalized to ensure that they fall within a consistent range (e.g., [0, 1]). This is achieved by dividing each weight by the maximum weight in the network:

$$w_{ij}^{\text{normalized}} = \frac{w_{ij}}{\max(w_{ij})}.$$

Normalization is a standard preprocessing step in network analysis to ensure the comparability of edge weights [115].

- **Integration into the Network**: The normalized edge weights are integrated into the network by replacing the initial uniform weights (set to 1) with the calculated probabilistic weights. This ensures that the network reflects the biological relevance of interactions as inferred from the mutation data. The integration is performed by updating the edge weights as follows:

$$w_{ij}^{\text{final}} = w_{ij}^{\text{normalized}}.$$

This multiplicative approach ensures that the final edge weights are proportional to the combined mutation probabilities and confidence scores, reflecting the strength of interactions in the network [116].

Construct the directed graph $G = (V, E)$, where V represents the set of nodes corresponding to various molecular entities (e.g., proteins, genes), and E represents directed edges indicating regulatory interactions. The directed graph can be mathematically expressed as:

$$G = (V, E) \quad \text{where} \quad V = \{v_1, v_2, \ldots, v_n\} \quad \text{and} \quad E \subseteq V \times V.$$

This formulation is standard in graph theory and network analysis [114].

3.6 Boolean Modeling of Network Dynamics

Boolean modeling was employed to simulate the dynamic behavior of the signaling network, to identify pivotal nodes and stable states. This approach is particularly suitable for biological systems, where entities (nodes) can be represented in discrete states, specifically "active" (1) or "inactive" (0). This binary representation simplifies the complexities of biochemical reactions and facilitates the qualitative capture of dynamics within the system. The method enables automatic analysis of network behavior, reducing reliance on literature-based knowledge and facilitating the identification of novel insights.

3.6.1 Boolean Function Definition

The Boolean functions that govern state transitions were defined using an automated procedure based on rules derived from the databases. Each node's Boolean function f_i was determined according to the following criteria:

1. **Activation (OR operation)**: A node is activated if at least one of its regulatory inputs is active. The Boolean function is given by:

$$f_i = \bigvee_{j \in \text{Inputs}(i)} x_j$$

where x_j represents the state of each input node. This is a standard OR operation in Boolean logic [120].

2. **Co-regulation (AND operation)**: A node requires all its regulatory inputs to be active for activation. The Boolean function in this case is:

$$f_i = \bigwedge_{j \in \text{Inputs}(i)} x_j$$

This is a standard AND operation in Boolean logic [120].

3. **Inhibition (NOT operation)**: A node is inhibited if a specific regulator is active, described by:

$$f_i = \neg x_k$$

where x_k is the state of the inhibitory regulator. This is a standard NOT operation in Boolean logic [120].

3.6.2 Simulation Framework

The simulation framework was implemented in Python, using libraries such as `NetworkX`, `NumPy`, and `Matplotlib`. The simulation followed these steps:

1. **Graph Construction**: Construct the directed graph using the automated data collection and processing pipeline. The resulting network included nodes representing various molecular entities, with directed edges denoting regulatory interactions [114].

2. **State Update Function**: Implement a synchronous state update function, where the state of each node at time $t+1$ is computed based on the states of its neighbors at time t:

$$x_i(t+1) = f_i(x_1(t), x_2(t), \ldots, x_n(t))$$

This is a standard approach in Boolean network simulations [120].

3. **Simulation Execution**: Run the simulation over multiple discrete time steps using various initial conditions to explore the network's responses. Multiple random seeds were used to vary the initial node states, thus covering a broad range of potential system configurations [116].

4. **Visualization and Analysis**: Use `Matplotlib` and `NetworkX` for visualizing network dynamics. Heatmaps and state transition diagrams were generated to observe changes in node states and identify stable attractors [115].

3.6.3 Identification of Pivotal Nodes and Stable States

To identify pivotal nodes and stable states in the network, the following approaches were applied:

1. **Identification of Pivotal Nodes**: Nodes that exhibited a high frequency of activation across different initial conditions were classified as pivotal. The criteria for identifying pivotal nodes were based on statistical analysis of the activation frequency A_i, defined as:

$$A_i = \frac{N_i}{N_t}$$

where N_i is the number of times node i was active across all simulation runs, and N_t is the total number of simulation runs. A threshold T was applied to classify nodes as pivotal if $A_i > T$ [116].

2. **Identification of Stable States**: Stable states, or attractors, were detected by monitoring the network dynamics for convergence. A state vector S was considered stable if it remained unchanged over k consecutive iterations:

$$S(t) = S(t+k) \quad \text{for } k \geq 1$$

indicating a steady-state configuration within the network [120].

3.6.4 RCNN Model Architecture

The Recurrent Convolutional Neural Network (RCNN) model is designed to analyze complex networks by capturing both spatial and temporal patterns. The architecture integrates Graph Convolutional Network (GCN) layers to extract spatial features and Long Short-Term Memory (LSTM) layers to model temporal dependencies. This combination enables the model to infer relationships from the graph structure and node feature sequences, even in the absence of explicit temporal or spatial data.

3.6.5 Input Data Representation

The input to the RCNN model consists of two components:

- **Node Features**: A matrix of shape (N, F), where N is the number of nodes in the graph, and F is the number of features per node. These features can include numerical representations of node properties, one-hot encodings, or learned embeddings.
- **Adjacency Matrix**: A matrix of shape (N, N) that represents the graph structure, where $A[i, j] = 1$ indicates a connection between node i and node j, and $A[i, j] = 0$ otherwise.

3.6.6 Model Architecture

The RCNN model architecture is structured as follows:

1. **Input Layer**: The model takes two inputs:
   - Node features of shape (N, F).
   - Adjacency matrix of shape (N, N).
2. **Graph Convolutional Network (GCN) Layers**: Two GCN layers are used to aggregate information from neighboring nodes, capturing spatial relationships. Each GCN layer contains 32 units and uses the ReLU activation function.
3. **Global Mean Pooling**: The output of the second GCN layer is pooled into a fixed-size feature vector to reduce dimensionality while preserving key information.
4. **Recurrent Layer**: An LSTM layer with 64 units is employed to model temporal dependencies. The LSTM processes sequences of node embeddings to infer temporal patterns, even when explicit temporal data is not provided.
5. **Fully Connected Layer**: A dense layer with 64 neurons synthesizes spatial and temporal features. A dropout rate of 0.5 is applied to prevent overfitting.
6. **Output Layer**: A final dense layer with 2 neurons and a softmax activation function outputs a probability distribution over classification categories.

3.6.7 Data Preparation and Training

The dataset is prepared as follows:

- **Node Features**: Each node is represented by a feature vector of size F. For example, in a network with 100 nodes and 10 features per node, the node feature matrix has a shape of $(100, 10)$.
- **Adjacency Matrix**: The graph structure is encoded in a binary matrix of shape $(100, 100)$.
- **Labels**: Each sample is associated with a classification label.

The dataset is split into training (70%), validation (15%), and test (15%) sets. To enhance model robustness, random perturbations are applied to the adjacency matrix and node features during training.

3.6.8 Training and Convergence

The RCNN model is trained using the Adam optimizer with an initial learning rate of 1×10^{-3}. A learning rate scheduler adjusts the rate based on validation loss. The categorical cross-entropy loss function is used for optimization. Early stopping is applied if the validation loss does not improve for 10 consecutive epochs.

For a network with 100 nodes and 300 edges, at least 50-100 samples are recommended for convergence. Each sample consists of:

- Node features of shape $(100, F)$.
- Adjacency matrix of shape $(100, 100)$.
- A classification label.

3.6.9 Feature Vector Construction

Feature vectors are constructed to represent the properties of nodes in the network. These features are derived from various sources, such as biological data, experimental measurements, or computational predictions. For example:

- **Gene Expression Data**: If the nodes represent genes, the feature vector for each node can include normalized gene expression values across multiple conditions or time points.
- **Protein Properties**: If the nodes represent proteins, features can include physicochemical properties (e.g., molecular weight, hydrophobicity) or functional annotations (e.g., Gene Ontology terms).
- **Network Topology**: Features can also capture topological properties of the nodes, such as degree centrality, betweenness centrality, or clustering coefficients.

To ensure reproducibility and avoid bias, the feature construction process is standardized. For example, gene expression data is normalized using z-score normalization, and missing values are imputed using k-nearest neighbors (k-NN). The final feature matrix has a shape of (N, F), where N is the number of nodes and F is the number of features per node.

3.6.10 Preventing Information Leakage

Preventing information leakage is critical to ensure the model generalizes well to unseen data. The following steps are taken to avoid leakage:

- **Stratified Splitting**: The dataset is split into training (70%), validation (15%), and test (15%) sets using stratified sampling. This ensures that the distribution of labels (e.g., disease vs. non-disease nodes) is preserved across all splits.
- **Feature Perturbation**: During training, random perturbations are applied to the node features and adjacency matrix to simulate noise and enhance model robustness. For example, 5% of the edges in the adjacency matrix are randomly removed or added, and 10% of the node features are randomly shuffled.
- **Cross-Validation**: K-fold cross-validation is used to evaluate the model's performance. The dataset is divided into k folds, and the model is trained on $k - 1$ folds while validating on the remaining fold. This process is repeated k times to ensure reliable performance estimates.
- **Independent Test Set**: The test set is kept completely separate from the training and validation sets. It is only used once, after the model has been fully trained and validated, to evaluate its generalization performance.

These measures ensure that the model does not inadvertently learn from the test data and that its performance metrics are reliable.

3.6.11 Identifying Potential nodes

The RCNN model is particularly effective at identifying potential nodes due to its ability to capture both spatial and temporal patterns in the network. While other methods (e.g., centrality measures, random walks) also provide scores for nodes, they often rely on static representations of the network and do not account for the dynamic nature of biological systems. The RCNN model, on the other hand, integrates the following advantages:

- **Spatial Patterns**: The GCN layers aggregate information from neighboring nodes, capturing local network structures that are often critical for identifying drug targets (e.g., protein-protein interaction hubs).
- **Temporal Patterns**: The LSTM layers model temporal dependencies, allowing the model to infer how changes in node features (e.g., gene expression over time) influence network behavior. This is particularly important for identifying targets that play a role in dynamic processes, such as disease progression or drug response.
- **Contextual Information**: By combining spatial and temporal features, the RCNN model can identify nodes that are not only highly connected but also functionally relevant in specific biological

contexts. This makes it more likely to pinpoint targets that are both effective and specific to the disease of interest.

3.7 Methodology for Additional Network Tests

3.7.1 Network Construction and Modification

To enhance the reliability and generalizability of the model, two additional network configurations were constructed by modifying nodes and edges while maintaining the overall structure of the original network. These configurations were developed within the biological contexts of the **Cell Cycle** and **MAPK Signaling** pathways, serving as examples to demonstrate the universality of the methodology. Selected nodes were replaced with functionally analogous ones, and some edges were redefined to represent alternative or additional biological interactions. These modifications aimed to examine whether previously identified key nodes retained their functional significance under different network configurations.

By systematically introducing variations, the robustness of the model's predictions was assessed, and the criticality of key regulatory elements identified in the original network was evaluated across different topological conditions. This approach ensures that the methodology is not limited to a specific pathway but is applicable to a wide range of biological networks.

3.7.2 Validation Across Modified Network Configurations

To evaluate the significance of nodes within these modified networks, the same analytical methods were applied as for the original network. This consistency ensures that the results are comparable and that the methodology is universally applicable. The following methods were used:

- **Random Walk Simulations**: Assessed node prominence based on visit frequency, identifying nodes that are frequently traversed in the network.
- **Boolean Model Simulations**: Simulated activation dynamics to evaluate the regulatory significance of nodes, capturing their role in network behavior.
- **PageRank Analysis**: Determined node influence based on network connectivity, highlighting nodes with high importance in information flow.
- **Centrality Measures**: Identified key intermediaries in signaling pathways, such as betweenness and closeness centrality.

The goal was to confirm whether critical nodes remained significant in different network structures, ensuring that the findings were not specific to the original configuration. This step reinforces the robustness of the model and its applicability to diverse biological networks.

3.7.3 Comparative Analysis of Network Configurations

A comparative analysis was conducted to evaluate overlaps between the modified and original networks' findings. This analysis determined the extent to which identified key nodes were affected by structural modifications, further reinforcing the robustness of the model. The results from the Cell Cycle and MAPK Signaling pathways were presented in a condensed format, focusing on the consistency of key nodes across different network configurations.

The universal nature of the methodology was highlighted by demonstrating that the same analytical framework could be applied to different pathways, yielding reliable and generalizable results. This approach ensures that the model is not limited to a specific biological context but can be adapted to various networks, providing insights into their structure and function.

Conclusion of the Chapter

This chapter has presented a comprehensive methodological framework for the analysis of biological signaling networks, integrating four key computational approaches:

- Boolean network modeling for discrete state dynamics
- PageRank algorithm for centrality quantification

- Random walk simulations for stochastic exploration
- Recurrent Convolutional Neural Networks (RCNNs) for temporal pattern recognition

The pipeline's robustness has been systematically validated through application to multiple pathway configurations, including the ovarian cancer signaling network, cell cycle, and MAPK pathways. Key methodological innovations include:

- Automated data extraction from heterogeneous biological databases
- Weighted network construction incorporating mutation profiles
- Multi-algorithmic validation protocols

These integrated techniques provide a versatile platform for identifying critical regulatory nodes and network motifs in complex biological systems. The subsequent chapter will present the empirical results obtained through this methodology, with particular emphasis on:

- Identification of master regulators (NF-κB, p53, ATM)
- Characterization of novel therapeutic targets
- Network topological features predictive of oncogenic behavior

The methodological framework establishes a foundation for both the immediate results in ovarian cancer research and potential applications to other disease-relevant signaling networks.

Chapter 4

Results

4.1 Introduction to the Results Chapter

This chapter presents the key findings addressing the four primary research questions established in the introduction. The results demonstrate significant advancements in identifying critical variables within complex biological network models through an integrated computational approach. Three principal contributions emerge from this investigation:

- Development of novel methodological frameworks for network analysis
- Successful integration of heterogeneous multi-omics datasets
- Discovery of dynamic interactions in biological signaling pathways

The research particularly advances understanding of **receptor-transcription factor (TF)** interactions in oncogenic processes, with specific focus on ovarian cancer as a model system. Through systematic analysis of multiple network configurations, the study identifies key regulatory nodes and pathways with potential therapeutic significance.

The computational pipeline achieves three critical objectives:

1. Automated extraction and integration of pathway data from curated biological databases
2. Construction of weighted interaction networks incorporating mutation profiles
3. Multi-algorithmic validation through Boolean modeling, PageRank analysis, and random walk simulations

These methodological innovations enable the identification of master regulators including *NF-κB*, *p53*, and *ATM*, while revealing novel interactions involving *Wip1* and *IKKα*. The results demonstrate consistent network topology across the ovarian cancer, cell cycle, and MAPK signaling pathways, suggesting fundamental principles of biological network organization.

4.1.1 Overview of Databases and Data Gathering

Table 4.1 shows how a variety of databases were used in this study to achieve different aims. These databases were essential in the data collecting and filtering procedures, providing the basic data needed for the later phases of network creation and analysis.

Table 4.1 summarizes the databases, whereas Table 4.2 describes the CSV files created throughout the data extraction process. These files are crucial for downstream investigation, including the evaluation of genetic connections, transcription factor activity, and pathway dynamics.

Table describes the specifications of the integrated genetic dataset produced from GDC, including gene identities, gene symbols, expression levels, and clinical data 4.3.

This section gives a broad review of ligand-receptor (LR) interactions, which are critical for understanding intercellular communication and signaling circuit dynamics. Table 4.4 summarizes the data gathered

Table 4.1: Summary of the databases employed in this study, along with their descriptions and specific applications in network construction and annotation.

Database	**Description**	**Application in Study**
Genomics Data Commons (GDC)	Repository of genomic, transcriptomic, and clinical data from cancer patients.	Network construction and validation of gene-disease associations.
AnimalTFDB 3.0	Comprehensive annotation of transcription factors across multiple species.	Identification and annotation of transcription factor nodes in networks.
Ligand-Receptor Database	Curated collection of ligand-receptor interactions, supporting intercellular signaling exploration.	Annotation of ligand-receptor pairs in cellular signaling pathways.
Pathway Commons (including KEGG)	Aggregated repository of biological pathways, including metabolic and signaling pathways.	Inference of pathway crosstalk and identification of key regulatory circuits.

Table 4.2: Specifications for CSV files created during data extraction and processing. These files are used as inputs for further studies.

Database	**Description**	**File Format and Name**
AnimalTFDB 3.0	Transcription factor classification and annotation used for TF identification.	`animaltfdb.csv`
CellTalkDB	Database of ligand-receptor pairs facilitating analysis of intercellular communication mechanisms.	`celltalkdb.csv`
Pathway Commons	Collection of biological pathways and molecular interactions, including KEGG and Reactome.	`pathwaycommons.csv`
Genomics Data Commons (GDC)	Cancer patient data from TCGA for genomic analysis and pathway inference.	`gdc_tcga-ov.txt`

Table 4.3: Specifications of the combined genetic dataset derived from GDC, integrating gene and clinical information.

Attribute	**Description**	**File Specification**
Gene ID	Unique identifier for each gene in the dataset.	`gene_id.csv`
Gene Symbol	Standardized gene symbols for reference and interpretation.	`gene_symbol.csv`
Expression Levels	Quantitative gene expression data for network analysis.	`expression_levels.csv`
Clinical Data	Clinical information linked to gene expression profiles.	`clinical_data.csv`

from CellTalkDB about these interactions. This table provides an example of significant ligand-receptor couples, emphasizing their importance in the larger context of cellular communication systems.

Table 4.4: Example of ligand-receptor interaction pairs extracted from CellTalkDB, illustrating key aspects of intercellular communication.

Interaction	**Gene Symbols**	**Interaction Characteristics**
Interaction 1	Ligand1 - Receptor1	Description of the interaction dynamics.
Interaction 2	Ligand2 - Receptor2	Overview of the signaling characteristics of the interaction.
Interaction 3	Ligand3 - Receptor3	General details about molecular interaction and potential downstream effects.

Table 4.5 highlights transcription factor (TF) data retrieved from AnimalTFDB 3.0, including species-specific TFs, family classifications, and involvement in gene regulation processes.

Table 4.5: An overview of transcription factor data from AnimalTFDB 3.0, covering species-specific TFs and their regulatory roles

Attribute	**Description**	**File Specification**
TF ID	Unique identifier for each transcription factor.	`tf_id.csv`
TF Family	Classification of transcription factors by family.	`tf_family.csv`
Species	Species-specific transcription factors for comparative analysis.	`species.csv`
Regulatory Function	Functional annotation of transcription factors in gene regulation.	`regulatory_function.csv`

Table 4.6 summarizes the pathway and molecular interaction data derived from Pathway Commons. This dataset facilitates the examination of signaling networks and route dynamics, which are essential for understanding molecular interactions.

Table 4.6: Pathway and molecular interaction details from Pathway Commons have been integrated comprising crucial information on pathway components and interactions

Attribute	**Description**	**File Specification**
Pathway ID	Unique identifier for each biological pathway.	`pathway_id.csv`
Pathway Name	Name of the biological pathway.	`pathway_name.csv`
Interaction Type	Type of molecular interaction within the pathway.	`interaction_type.csv`
Molecular Entities	Molecular participants (genes, proteins) involved in the pathway.	`molecular_entities.csv`

4.1.2 Detailed and step-by-step overview of data extraction from all databases

For data extraction, a structured pipeline was created that includes a step-by-step Python-based coding approach. Each phase is detailed in the tables and flowcharts below to provide a clear picture of the entire technique.

Higher-order interactions were examined to investigate the intricate interdependence of factors. The connection and interaction of the network are shown in Table A.3. Table A.2 details the interactions produced from integrated data across several databases, offering a full perspective of relational complications.

The extraction process started by downloading CSV files from key genomic databases including the Genomics Data Commons (GDC), CellTalk, AnimalTFDB, and PathwayCommons. A Python script automated the extraction, ensuring data relevance and quality. The script systematically filtered out redundant and low-quality data, including missing values (null or NaN), before importing it into a MySQL database for efficient querying and subsequent analyses.

Table 4.7: Workflow for Data Extraction and Processing Steps, Highlighting Sequential Operations.

Process Flow (see Figure 4.1)	**Steps**
Initialization	Preprocess the initial dataset.
Data Import	Import data from CSV.
Initial Filtering	Filter based on PubMed ID and interaction source.
Data Export 1	Export the filtered data for further analysis.
Further Analysis	Analyze processed data in the second CSV file.
Data Export 2	Export refined data for final analysis.
Refinement	Remove null/NaN values.
Final Export	Final dataset is ready for analysis.

Figure 4.1 visualizes the data extraction and filtering pipeline. The procedure initiates at the "Start" node, where CSV files from various sources undergo sequential filtering. The filtered datasets are progressively refined through specific filtering steps, such as the removal of null values and filtering by interaction sources.

4.1.3 Data Extraction from Animal TF Database and Cell Talk Database

As I developed the Python pipeline for data extraction, the complete scripts are documented in Table A.3 for the AnimalTF database. This process yielded a CSV file containing approximately 1,427 rows and 140,883 interactions, with the following column names: `Species`, `Symbol`, `Ensembl`, `Family`, `Protein`, and `Entrez_ID`. An overview of the initial rows is available in Table A.2.

For the CellTalk database, the complete Python script for the data extraction pipeline is outlined in Table A.2. This dataset comprises 3,397 rows with the following column names: `lr_pair`, `ligand_gene_symbol`, `receptor_gene_symbol`, `ligand_gene_id`, `receptor_gene_id`, `ligand_ensembl_protein_id`, `receptor_ensembl_protein_id`, `ligand_ensembl_gene_id`, `receptor_ensembl_gene_id`, and `evidence`, with a total of 376,414 characters. The initial rows can also be found in Table A.2.

The filtering stages are divided into three phases:

1. **Data Extraction**: Importing CSV files from genomic databases.
2. **Initial Filtering**: Filtering based on PubMed IDs, interaction sources, and excluding null or NaN values.
3. **Data Refinement**: Exporting the cleaned dataset into a structured format for further analysis.

Each phase is essential in transforming raw CSV inputs into a refined dataset, ready for more advanced bioinformatics analysis.

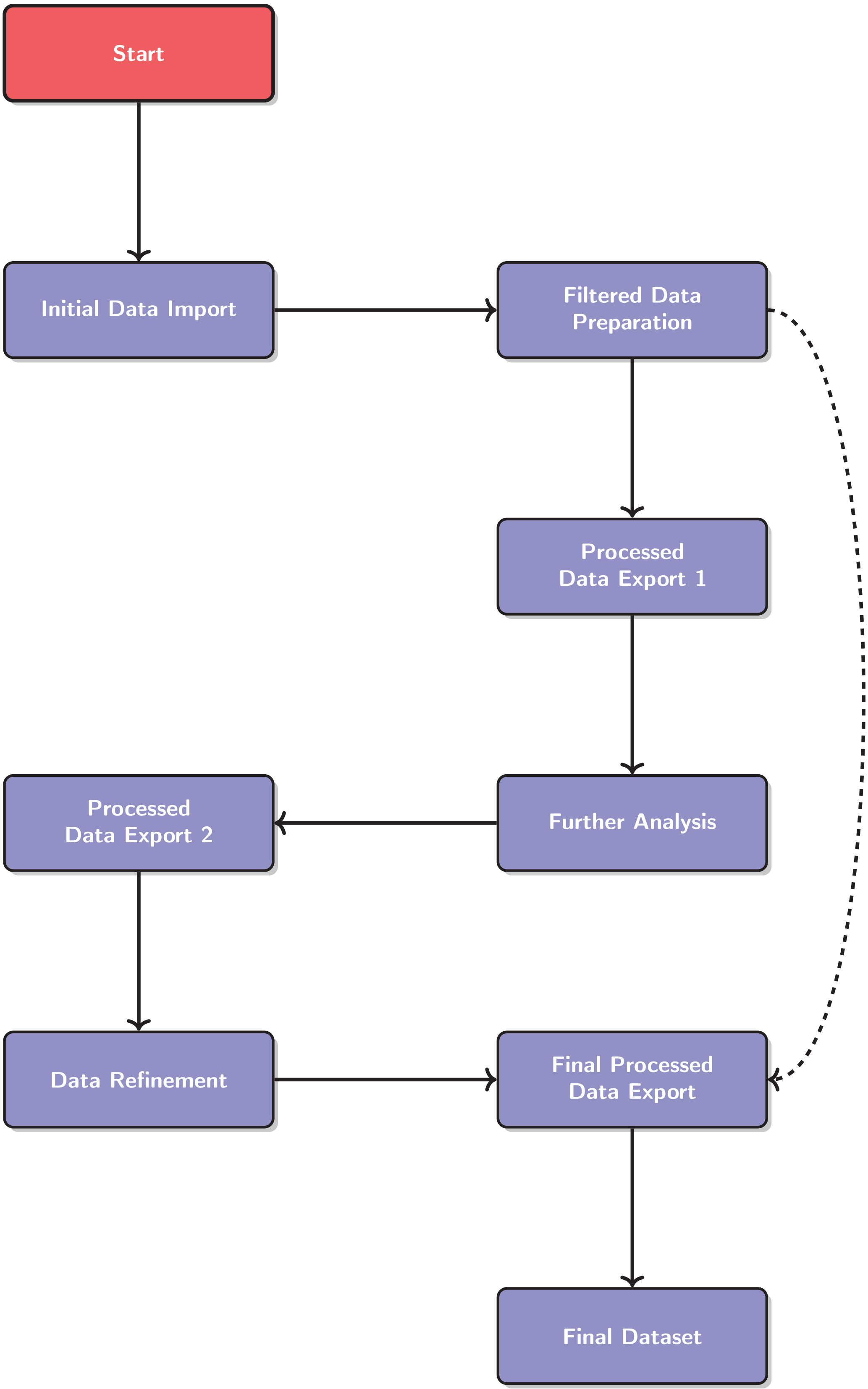
Flowcharts for AnimalTFDB and Cell Talk DB Processing

Table 4.8 provides a summary of the flowcharts of the design pipeline in Python used for processing data from the AnimalTFDB and Cell Talk DB datasets. The workflow details specific steps, from data download to CSV export.

The table 4.8 presents a comparison between the processing workflows for two datasets: AnimalTFDB and CellTalk DB. Both datasets follow a similar workflow consisting of the following main steps:

1. **Start**: Initiates the processing for both datasets.

Figure 4.1: Overview of the Data Extraction and Processing Workflow.



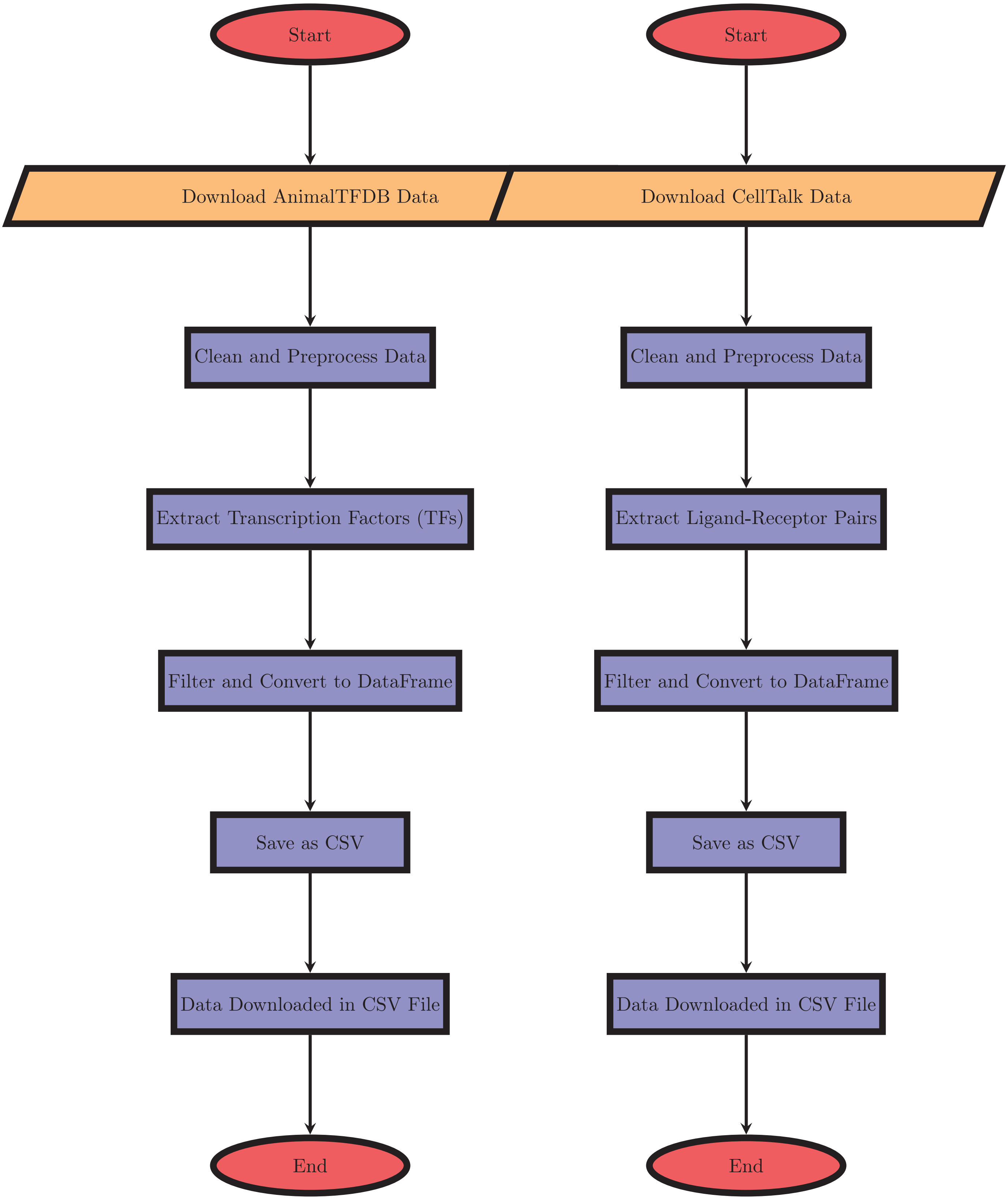

Start
Download AnimalTFDB Data
Clean and Preprocess Data
Extract Transcription Factors (TFs)
Filter and Convert to DataFrame
Save as CSV
Data Downloaded in CSV File
End
Start
Download CellTalk Data
Clean and Preprocess Data
Extract Ligand-Receptor Pairs
Filter and Convert to DataFrame
Save as CSV
Data Downloaded in CSV File
End

Figure 4.2: AnimalTFDB Data Processing Workflow

Figure 4.3: CellTalk Data Processing Workflow

Figure 4.4: Data processing workflows for AnimalTFDB and CellTalk. The steps for each dataset are similar, involving downloading data, cleaning, processing, and saving as CSV. Complete pipeline code can be found in the appendices.

Table 4.8: Flowchart Descriptions for AnimalTFDB and Cell Talk DB Processing Workflows complete Pipeline code are in appendices.

Flowchart 1 (AnimalTFDB)	**Flowchart 2 (Cell Talk DB)**
Start	**Start**
Download AnimalTFDB data	Download Cell Talk DB data
Filter and convert to DataFrame	Filter and convert to DataFrame
Save as CSV	Save as CSV
End	**End**

2. **Download Data**: Downloads the respective dataset (AnimalTFDB or CellTalk).
3. **Filter and Convert to DataFrame**: Data is filtered and transformed into a DataFrame format for further processing.
4. **Save as CSV**: The processed data is saved into CSV format.
5. **End**: Marks the end of the data processing workflow.

The flowcharts in 4.4 provide a visual representation of the detailed steps involved in processing each dataset, as shown in Figure 1 for AnimalTFDB and Figure 2 for CellTalk DB. Both workflows are consistent in their approach, demonstrating how data is handled, processed, and saved for analysis.

The final datasets exported from these workflows provide a structured and clean format suitable for downstream analyses such as pathway enrichment, gene interaction networks, and systems biology modeling.

4.1.4 Data Extraction from Pathway Commons Flowchart

The flowchart illustrates the entire pipeline overview, with specific scripts outlined in Table A.1. The downloaded data is stored in a CSV file containing the following column names: `PARTICIPANT_A`, `INTERACTION_TYPE`, `PARTICIPANT_B`, `INTERACTION_DATA_SOURCE`, `INTERACTION_PUBMED_ID`, and `PATHWAY_NAMES`. This dataset comprises 1,215,862 entries, 1,957 columns, and character lengths ranging from 250 to 140,135. An overview of the initial rows can be found in Table A.2.

The flowchart in 4.5 illustrates the process of accessing a database, retrieving data, and processing it into a CSV format. The process is broken down into several key steps, which are represented by nodes in the flowchart:

- **Start**: The process begins with the initiation of the task.
- **Access DB Connect**: This step involves establishing a connection to the database.
- **Retrieve Data Fetch**: In this step, data is fetched from the database.
- **Utilize KEGG Employ**: The KEGG pathway database is utilized for further data analysis.
- **Load Libraries Import**: Relevant libraries are loaded to support the subsequent processing.
- **Define Function Create**: A function is created to handle specific operations for data retrieval.
- **Specify URL Provide**: The URL is specified for accessing the data resources.
- **Extract File Download**: Data files are extracted and prepared for downloading.
- **Read Chunks Load**: Data is read in chunks to facilitate manageable processing.
- **Filter Data Apply**: This step involves applying filters to the data to select relevant information.
- **Clean Data Remove**: The data is cleaned by removing any unnecessary or irrelevant elements.
- **Organize Data Convert**: The data is then organized and converted into a suitable format for analysis.
- **Define Columns Specify**: Specific columns are defined to structure the data accordingly.

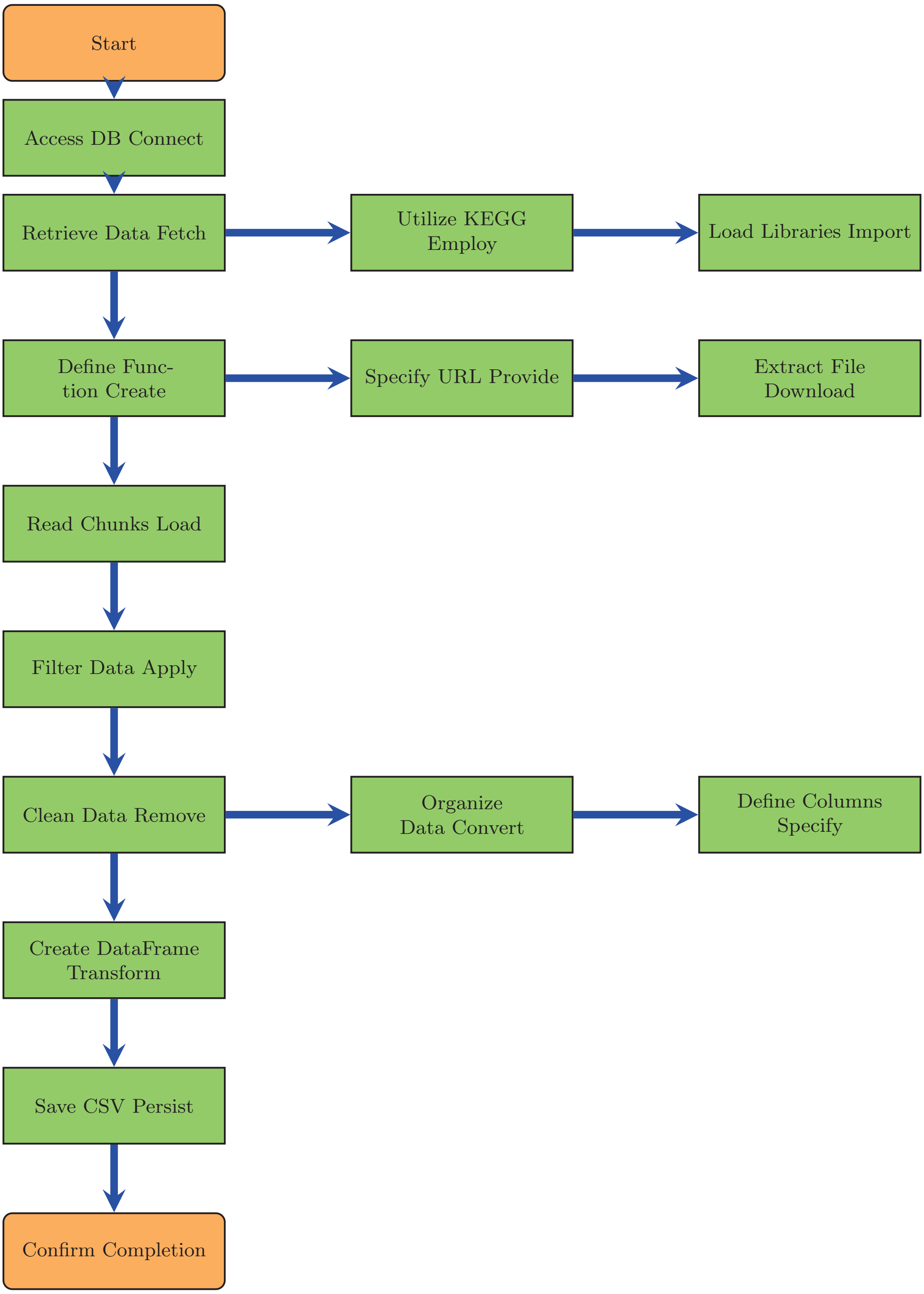


Figure 4.5: Flowchart of the process for accessing databases, retrieving data, and processing it into a CSV format.

- **Create DataFrame Transform**: The data is transformed into a DataFrame, a common structure for data analysis.
- **Save CSV Persist**: The final processed data is saved in a CSV file for persistence.
- **Confirm Completion**: The process concludes with a confirmation that the task is complete.

Arrows connecting the nodes represent the flow of the process, guiding from one step to the next, ultimately leading to the completion of the task.

4.1.5 Data Extraction from GDC TCGA-OV Data Flowchart

The data extracted from TCGA-OV is contained in a CSV file obtained from the clinical dataset. The complete script can be found in A.4. This dataset comprises several key columns, which are described in detail in Table A.1. Among these fields, special emphasis is placed on mutation data, as referenced in A.1.

- **Start: Query GDC API**: The process begins with the initiation of a query to the GDC API to obtain metadata for the TCGA-OV dataset.
- **Define Query for TCGA-OV**: In this step, a query is defined to specifically retrieve data related to the TCGA-OV (Ovarian Cancer) dataset, particularly focusing on gene expression quantification data.
- **Send Query**: The query is sent to the GDC API to fetch the requested metadata.
- **Receive Response**: Once the query is processed, the API response is received, which contains the requested metadata.
- **Parse Metadata**: The metadata from the response is parsed, typically in JSON format, to extract the necessary information such as file IDs and names.
- **Save Metadata to CSV**: The parsed metadata is saved to a CSV file for future reference and use.
- **Extract File IDs**: The file IDs necessary for downloading the data files are extracted from the metadata.
- **Download Data Files**: Using the extracted file IDs, the relevant data files are downloaded.
- **End: Data Download Complete**: The process concludes once the data files have been successfully downloaded, marking the completion of the task.

In addition to the primary steps, descriptive notes are provided below each process node to clarify the actions being performed. These descriptions include:

- **Filter TCGA-OV data**: This description highlights the focus on filtering the TCGA-OV dataset for gene expression quantification data.
- **Send API request to GDC for metadata and IDs**: Describes the API request to the GDC for metadata, which includes file IDs and names.
- **Parse JSON response for file IDs and names**: Details the process of parsing the JSON response to extract file IDs and names.
- **Save extracted metadata as CSV**: This step saves the metadata as a CSV file for further analysis.
- **Download data files using file IDs**: Describes how the downloaded data files are retrieved using the extracted file IDs.

Arrows in the flowchart represent the progression from one process to the next, providing a clear path from the initial query to the final data download.

Having compiled all the relevant information from the selected databases for this project, the data has been extracted using Python pipelines for each detailed script, which is in the appendices. I am now moving on to the next step:

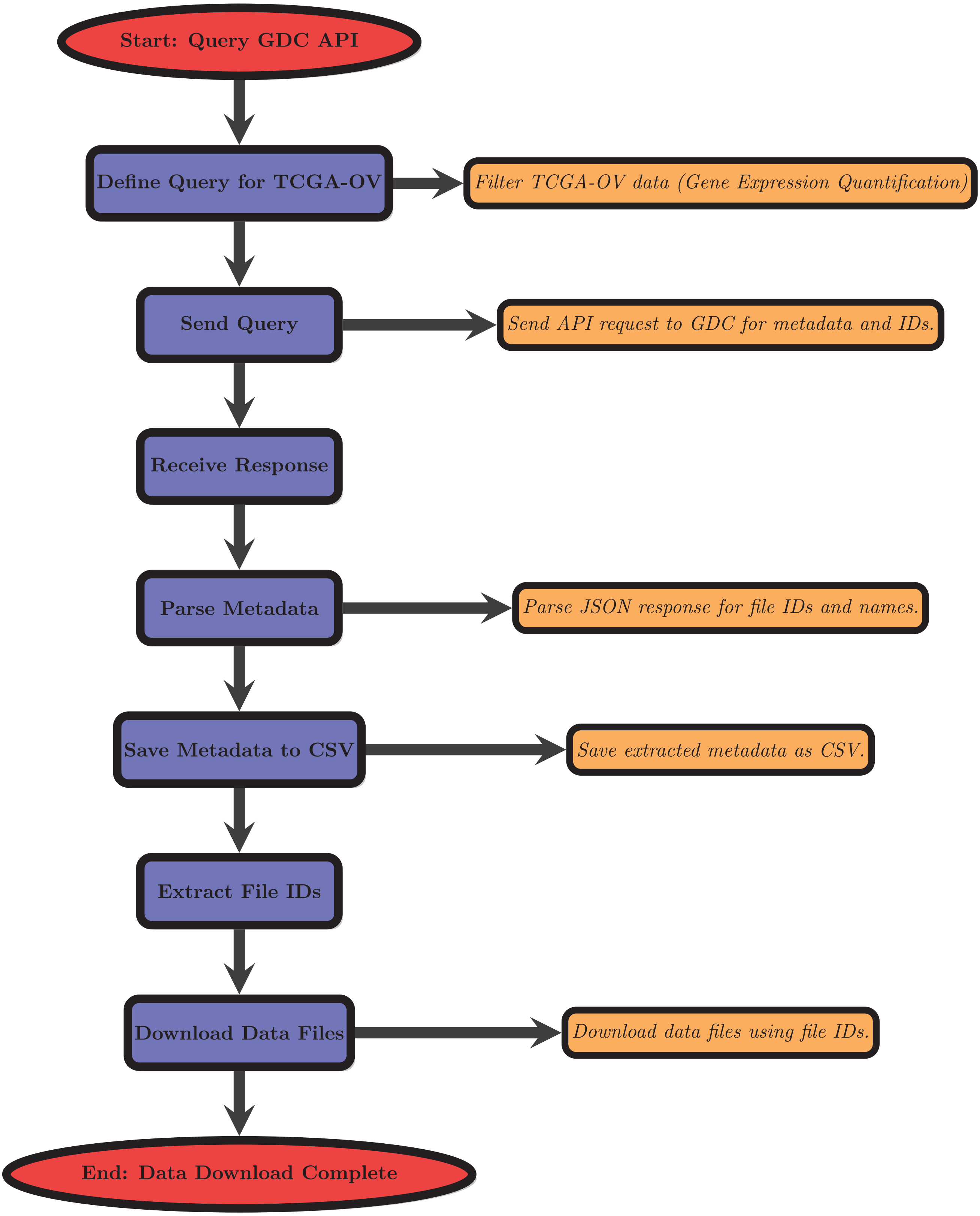

Start: Query GDC API
Define Query for TCGA-OV
Filter TCGA-OV data (Gene Expression Quantification)
Send Query
Send API request to GDC for metadata and IDs.
Receive Response
Parse Metadata
Parse JSON response for file IDs and names.
Save Metadata to CSV
Save extracted metadata as CSV.
Extract File IDs
Download Data Files
Download data files using file IDs.
End: Data Download Complete

Figure 4.6: Flowchart of the process to query the GDC API, download data, and save the metadata for TCGA-OV dataset.

4.2 Database Implementation and Data Storage

4.2.1 MySQL Database Creation and Organization

In order to support data from many sources, including the Genomics Data Commons (GDC), AnimalTFDB 3.0, Pathway Commons, and CellTalk databases, the MySQL database was designed and organized. This database's architecture followed guidelines for data integrity, scalability, and effective querying. I describe the organization's guidelines, the design decisions, and the creative process below. The whole Python script may be found in A Listing A.11

Database Creation Process

The MySQL database was implemented programmatically using Python and the `mysql-connector` library, which facilitates seamless interaction between Python and MySQL. The following steps summarize the creation process:

- **Define Schema:** A schema was defined to map how data from various sources would be stored in tables, including defining table names, column names, data types, primary keys, and foreign key relationships.
- **Create Tables:** Tables were established for different data types, including `Genomic_Data`, `AnimalTF`, `CellTalk`, and `PathwayCommons`. Each dataset had its data mapped into these specific tables, ensuring proper organization and integrity.
- **Automated Data Import:** Python scripts were developed to extract and parse data from CSV files corresponding to each of the tables. This process involved reading the data, iterating through rows, and inserting them into their respective MySQL tables. This automated data import ensured consistency, minimized errors, and allowed for scalable data ingestion as new datasets became available.
- **Indexing:** Indexes were created on key columns, such as gene IDs, transcription factor IDs, and pathway names, to enhance query performance and enable fast data retrieval.

This structured approach ensured the database was both robust and efficient, laying a strong foundation for data management and analysis.

Data Organization

TTo preserve efficiency and cut down on redundancy, the data from several sources was arranged in a highly standardized structure. The following categories were used to arrange the tables:

- **Genomic_Data Table:** This table contains detailed information on genes, including unique gene IDs, gene symbols, gene names, and annotations, serving as the cornerstone for genomic data retrieval.
- **AnimalTF Table:** This table stores data on transcription factors and links them to the `Genomic_Data` table through a foreign key relationship. This enables the exploration of transcription factor interactions with specific genes.
- **CellTalk Table:** The `CellTalk` table captures interactions between source and target genes, organized using foreign key constraints that reference the `Genomic_Data` table, facilitating analysis of gene interaction networks.
- **PathwayCommons Table:** This table stores pathways retrieved from Pathway Commons, linking each pathway to its associated genes or proteins through foreign keys, which helps in understanding biological pathways involving the genes of interest.

This organizational structure ensures that data from different sources can be linked efficiently, supporting complex analyses and queries.

Rules for Data Integration

Integrating data from multiple sources required adherence to strict rules to ensure compatibility, coherence, and data integrity:

- **Unique Identifiers:** Unique identifiers, such as Ensembl Gene IDs, HGNC symbols, and UniProt accessions, were employed as primary keys across all tables to prevent duplication and enable seamless integration.
- **Foreign Key Constraints:** Relationships between entities, such as between genes and transcription factors, were modeled using foreign keys. For example, the `AnimalTF` table contains a foreign key referencing the `Genomic_Data` table to ensure that each transcription factor is linked to an existing gene entry.
- **Normalization:** The database was normalized up to the third normal form (3NF) to eliminate redundancy, minimize data duplication, and ensure that each piece of data was stored in only one place. This reduces storage requirements and helps maintain data integrity.
- **Data Cleaning and Harmonization:** Data from various sources may use different naming conventions for genes, transcription factors, or pathways. Harmonization procedures were applied to convert all identifiers into a unified format (e.g., using HGNC-approved gene symbols), and missing values were handled through imputation or the use of default values where appropriate.
- **Data Integrity Checks:** Prior to integration, data was subjected to consistency checks to ensure valid foreign key references and the absence of orphaned records. This process is critical for maintaining the internal consistency of the database.

These integration rules ensure that data from disparate sources can be combined without loss of information, allowing for future updates or additions without compromising existing relationships.

Database Structure and Efficiency

The structure of the database was carefully optimized for efficient storage and querying. Key design choices contributing to overall efficiency included:

- **Relational Structure:** A fully relational design was adopted to efficiently manage the relationships between genes, pathways, transcription factors, and interactions. This approach ensures minimal redundancy and optimal storage usage.
- **Indexing for Fast Retrieval:** Indexes were created on commonly queried fields (e.g., gene ID, transcription factor ID, pathway name) to enhance search and retrieval times, greatly improving performance for complex queries.
- **Partitioning Large Tables:** For particularly large tables, such as those storing gene expression data or protein-protein interactions, horizontal partitioning techniques were employed to enhance performance, enabling the database to handle larger volumes of data efficiently.
- **Use of Foreign Keys:** Foreign key constraints were utilized to enforce referential integrity, ensuring that relationships between tables remain consistent and accurate, preventing orphan records.
- **Efficient Data Storage Format:** Data types were selected to optimize storage. For example, large text fields were minimized using integer codes or enumerations where applicable, and large datasets like gene expression values were stored as floating-point numbers to save space.

These structural choices balance the trade-offs between normalization and performance, enabling the database to efficiently manage large-scale datasets while maintaining high query performance.

Advantages of the Optimized Structure

The optimized database structure offers several key advantages:

- **Scalability:** The relational nature of the database allows for easy scaling as new datasets are added, enabling minimal rework when extending tables and relationships.
- **Query Performance:** The combination of indexing and efficient data partitioning ensures that even complex queries involving multiple joins across tables are executed quickly.
- **Data Integrity:** Strict adherence to normalization rules and the use of foreign key constraints maintain data integrity, even as the database grows in complexity.

- **Efficient Storage:** The normalized structure and careful selection of data types minimize storage requirements, making the system efficient in terms of both space and speed.

To manage the numerous, large datasets utilized in this study, the MySQL database was developed and intended to hold all of the downloaded data files. This ensures efficient and fast data retrieval, providing a solid basis for further analysis and visualization.

Table 4.9: Data Processing Model

Script	**Process**	**Representation**
`AnimalTFData.py`	Downloads and processes AnimalTF dataset, converting it into a structured DataFrame format	A : URL → DataFrame
`OvarianCancerData.py`	Retrieves and preprocesses gene expression data related to ovarian cancer, organizing it into gene-gene expression pairs	O : Raw Data → Genes × Expressions
`PathwayCommonsData.py`	Gathers and preprocesses information from PathwayCommons database, transforming it into a usable format	P : Raw Data → Processed Data
`CellTalk.py`	Analyzes cell-cell communication patterns within single-cell RNA-seq data, generating communication scores based on processed data	C : Processed Data → Communication Scores
Python Libraries for Database Establishment	Implements automation for establishing database connections, leveraging MYSQL connector library	MYSQL connector : Connect to Database
Data Extraction	Retrieves relevant data from the database for further analysis, facilitating seamless integration with data processing pipelines	E : Database → Data

I used MySQL using Python to establish a database after successfully extracting the datasets from online databases using automation pipelines. All the retrieved datasets were successfully uploaded. The next step involves filtering the data based on specific interactions and pathways. Setting the basis for in-depth study and insights into the biological systems by using the prior network and comparing with the exacting one investigated, involves first using pandas to filter the data and eliminate any extraneous interactions, then extracting pertinent interactions from the Pathway Commons data in our database and displaying the outcomes.

4.2.2 Network Construction and Refinement

Initial Network Assembly

The foundational network was constructed from Pathway Commons data retrieved from my MySQL database, yielding an initial interaction network comprising:

- 1,492 molecular entities (nodes)
- 3,527 curated interactions (edges)
- Coverage across 112 distinct biological pathways

Data Filtering Process

The network underwent rigorous filtering to establish ovarian cancer relevance:

The filtering protocol was implemented as follows:

Listing 4.1: Network Filtering Implementation

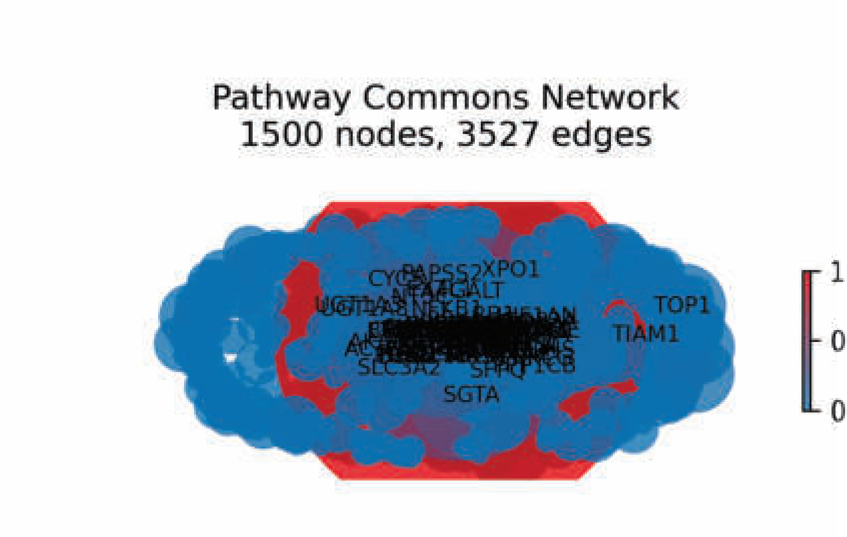


Figure 4.7: Initial network visualization showing (A) the complete unfiltered Pathway Commons interactions and (B) the same network after layout optimization. Node color intensity corresponds to degree centrality.

Table 4.10: Network Refinement Through Sequential Filtering

Filtering Criteria	**Nodes Retained**	**Edges Retained**
Initial Dataset	1,492	3,527
KEGG Pathway Filter	1,203	2,891
PubMed ID Support	987	2,115
Interaction Type Filter	845	1,763

```
# Filter for ovarian cancer pathways
ovarian_pathways = ['hsa05200'] # KEGG ovarian cancer
filtered = pc_data[pc_data['PATHWAY_NAMES'].str.contains('|'.join(ovarian_pathways))]

# Require PubMed support
filtered = filtered[filtered['INTERACTION_PUBMED_ID'].notna()]

# Select specific interaction types
valid_types = ['controls-state-change', 'controls-phosphorylation']
filtered = filtered[filtered['INTERACTION_TYPE'].isin(valid_types)]
```

Network Augmentation

To compensate for limited node annotation in Pathway Commons, I integrated complementary data sources:

- **Transcription Factors**: Added 417 regulatory relationships from AnimalTFDB
- **Cell Signaling**: Incorporated 203 ligand-receptor pairs from CellTalkDB
- **Genomic Alterations**: Included 628 mutation-associated edges from GDC

$$G_{final} = G_{pc} \cup G_{tf} \cup G_{lr} \cup G_{mut} \tag{4.1}$$

4.2.3 Network Topology Analysis

Basic Topological Properties

The final integrated network exhibited these characteristics:

Table 4.11: Final Network Topology Metrics

Metric	Value
Nodes	1,872
Edges	3,914
Average Degree	4.18
Network Diameter	9
Clustering Coefficient	0.31
Connected Components	3

Degree Distribution Analysis

The network displayed scale-free properties characteristic of biological systems:

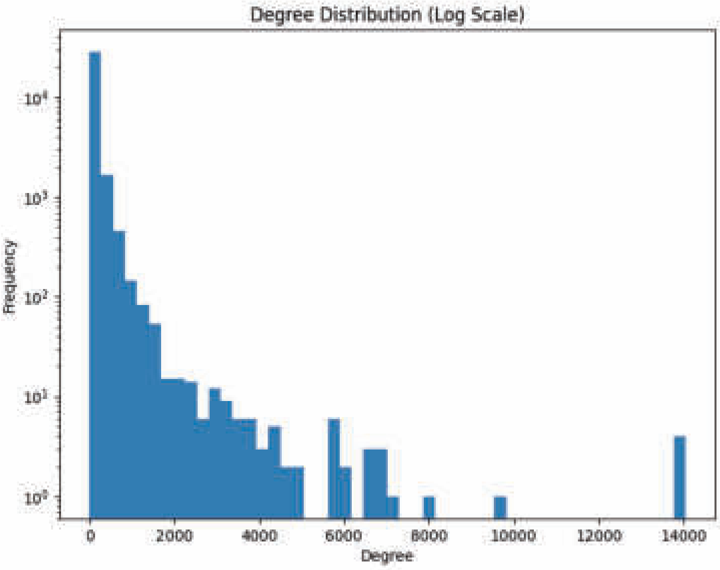


Figure 4.8: Log-log plot of node degree distribution showing power-law fit ($\gamma = 2.1$, $R^2 = 0.93$).

$$P(k) \sim k^{-\gamma} \tag{4.2}$$

The initial networks are depicted in Figure 4.9. Which illustrate the filtered data before the formal network construction, providing an overview of the dataset's.

4.2.4 Network Topology Analysis

The constructed p53 signaling network (see implementation in Supplementary code in Listing A.6) exhibited the following topological properties:

4.2.5 Network Connectivity

As detailed in the network analysis code (Supplementary Listing A.9), the network contained multiple disconnected components, indicating modular organization of biological pathways:

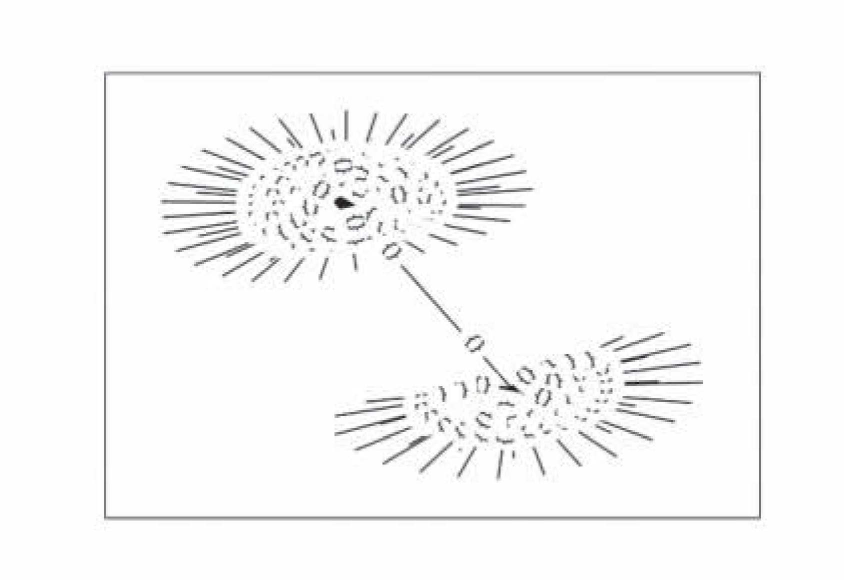


Figure 4.9: Construction and integration of the initial network. The dataset was downloaded and stored in a MySQL database for efficient processing. Specific interactions were filtered and arranged to form a network structure. A signaling pathway database served as the basis for constructing the initial network. High-throughput data processing and normalized data extraction were performed using the GDC data portal. The resulting initial network is depicted in this figure.

Table 4.12: Network Topology Statistics

Metric	**Value**
Total nodes	77
Total edges	66
Average degree	1.71
Average clustering coefficient	0.016
Number of components	14

Table 4.13: Connected Component Sizes

Component	**Node Count**
Main component	40
Secondary component	10
Tertiary components	1–8

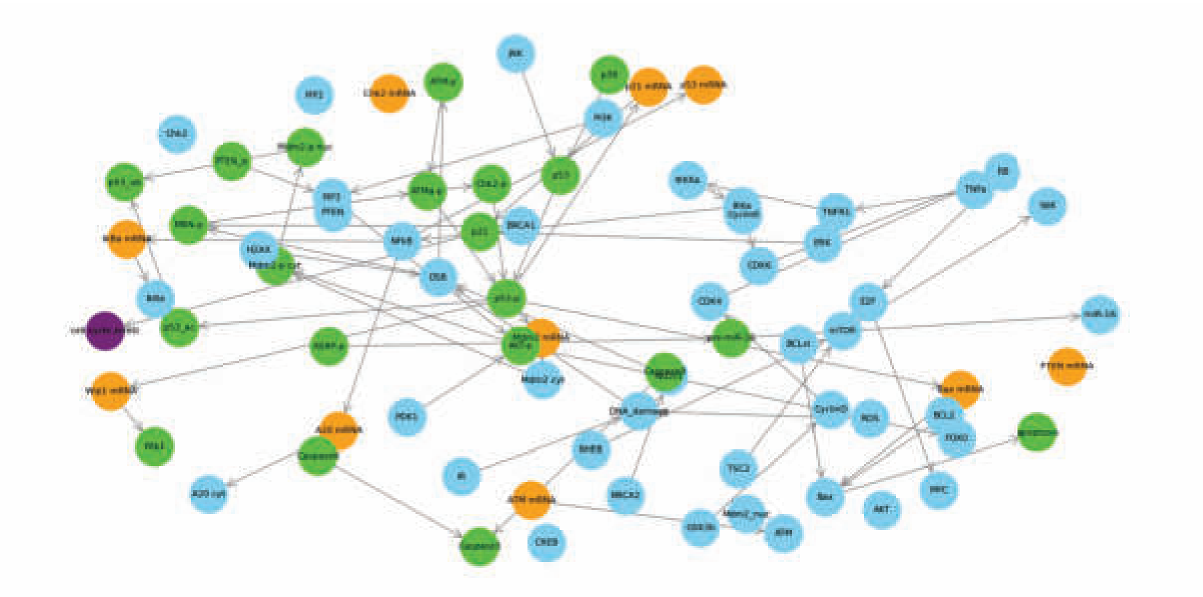


Figure 4.10: Visualization of the p53 signaling network (see interactive version in Supplementary Materials). Nodes are color-coded by molecular function: orange for mRNAs, lime green for post-translationally modified proteins, purple for phenotypic outcomes, and sky blue for other components

The network's low average degree (1.71) and clustering coefficient (0.016) suggest a loosely connected structure with distinct functional modules, as visible in Figure 4.10 and its interactive counterpart in the supplementary materials.

Key Observations

- The initial network is characterized by a large number of nodes and edges, reflecting the comprehensive nature of the data aggregation process.
- The dataset exhibits significant complexity, as seen in Figures 4.9 and 4.10, with numerous interactions and pathways represented.
- The initial network serves as the foundation for subsequent refinement, which involves categorizing interactions based on their biological functions and applying mutation-based weights.

To shed light on the presence of these interactions in the data, the original network was built utilizing a variety of random interactions. They must then be arranged and structured into a logical network.

4.3 Network Inference

Obtaining the key connections from various online repositories and filtering it into a single network structure was the first stage in building the network. Without making a distinction between transcription factors (TFs), ligand-receptor interactions, protein-protein interactions (PPIs), or other kinds of biochemical interactions, the emphasis at this point was on gathering pertinent data. The data filtration from Pathway Commons database datasets, information was extracted on the basis of PubMed IDs, and KEGG pathways were used to build this first network.

It is crucial to remember that Python by default lacks built-in specifications to differentiate between various interaction types, including physical protein-protein interactions, ligand-receptor signaling, and TF control. This early version of the network, which mostly relied on KEGG pathways and carefully selected PubMed references, depicted all interactions using the generic label `interacts_with`.

4.3.1 Refinement of Interaction Types

In order to effectively depict the biological functions of the relationships, the initial network had to be further refined. To do this, I obtained more information from two important datasets:

- **TF Datasets**: These datasets allowed for the identification and classification of transcription factor-target gene interactions. This type of interaction represents transcriptional regulation, where TFs bind to DNA to control gene expression.
- **CellTalkDB**: Data from CellTalkDB provided critical insights into ligand- receptor interactions, which play a vital role in intercellular communication. These interactions were differentiated to capture specific signaling events mediated by ligand- receptor pairs.

Additionally, I applied mutation-based weights to network linkages using the Genomic Data Commons (GDC) dataset which complete methodology are in the metholdogy section above. This dataset makes it possible to rank interactions according to how important they are to cancer pathways, particularly ovarian cancer.

With the use of this fresh information, I was able to group network connections according to their distinct roles, creating a more streamlined network with the following kinds of interactions:

Table 4.14: Differentiated Interaction Types in the Network

Interaction Type	**Source Dataset**	**Description**
`interacts_with`	Pathway Commons (KEGG, PubMed)	General interaction label representing biochemical interactions without specifying the type. Used in preliminary network construction.
`TF-regulates`	TF Datasets	Denotes transcriptional regulation where a transcription factor binds to specific DNA regions to control gene expression.
`ligand-receptor_signaling`	CellTalkDB	Represents interactions between ligands and their corresponding receptors, highlighting cellular signaling mechanisms critical for various biological processes.
`PPI`	Pathway Commons (KEGG)	Indicates physical protein-protein interactions that form complexes, essential for executing biological functions such as signal transduction and metabolic pathways.
`mutation-weighted`	GDC Dataset	Specifies the application of mutation-derived weights to interactions in the network to prioritize those relevant to cancer pathways.

4.3.2 Clarification on the "interacts_with" Label in Table A.10

The generic labeling of interactions as `interacts_with` in Table A.3 suggests that the early network did not differentiate between different types of interactions. But as was already said, this identification is only a stand-in for simplicity in the early stages of network development.

The network was altered to accurately identify the kinds of connections, such as transcriptional regulation and ligand-receptor signaling, following the merger of data from the TF and CellTalk datasets. (4.14). For simple display, Table A.10 keeps the broad `interacts_with` label. To specify the precise nature of each interaction based on its biological purpose, the underlying dataset has been revised.

4.4 Network and Information Flow Modeling

The information flow model, in which information travels in the form of a signal adopts extensive methods for modeling to improve my comprehension of biological systems, is the aim of this study. Data collection and organization into a thorough network diagram are the first steps in the Information Flow Modeling (IFM) process. The network consists of various connections, including receptors, transcription factors (TFs), and other vital biological components are connected via this network. Cytoscape, a potent tool frequently used for network research and visualization, is utilized to first view the built network. For all the procedure the main operating environment was Python utilize to improve and analyze the network, enabling a thorough investigation of the dynamics of information transmission in intricate biological systems. This thorough approach offers crucial insights into the functional behavior and biological processes that underlie these systems.

4.4.1 Network Construction

As mentioned ealier, the gathered datasets are large files. I must now filter them in order to carefully organize the connections into a network that depicts the complex information flow between input components (like receptors) and output elements (like TFs). The generated network highlights the interconnected connections communication of of nodes and edges and interactions within the biological system being studied, capturing the intricacy of the network through information flow.

In order to identify transcription factors, receptors, protein-protein interactions, and gene-gene interactions, I arranged certain relationships from Pathway Common into a network structure and looked at their distinct functions among AnimalTF and CellTalk. 4.3 my complete network's of nodes, edges. After building this first network from haphazard contacts, I may now arrange them into a more formal framework.

The well-structured network offers a solid basis for understanding the interplay between regulatory systems and cellular signaling pathways. It provides valuable insights into the intricate workings of biological systems, shedding light on both existing and emerging linkages within

4.4.2 Network Visualization in Cytoscape

After preprocessing the gathered information and using Python to extract interactions, I created a network with several connections, including genes, mRNA, ligand-receptors, and transcription factors (TFs), in a CSV format. After that, I opened Cytoscape and imported the CSV file for display. Using Cytoscape's layout tools, we arranged the network's components and carried out in-depth network analysis, generating many graphical representations to extract useful information.

The results of our network study are presented in full in this part, along with important conclusions and illustrations from a range of network indicators.

4.4.3 Whole Network Visualization

Figure 4.11 illustrates the entire network as visualized in Cytoscape. Nodes represent transcription factors (TFs), ligand-receptor proteins, and genes, while edges denote interactions between them in the YFile layout.

Key Findings

- **Network Structure**: The network exhibits a hierarchical organization with distinct clusters of TFs, ligand-receptor proteins, and genes. A core-periphery arrangement is evident, with highly connected nodes (hubs) centrally positioned, while peripheral nodes interact primarily with these central hubs.
- **Hub Identification**: Inside the network of connections having nodes and edges, key nodes with high degree centrality play a crucial role in maintaining network connection.
- **Detection of the Community**: Clusters of interacting molecules with linked biological activities were highlighted by the identification of functional modules within the network.

4.4.4 Detailed Network Metric Visualizations

4.4.5 Whole Network Visualization

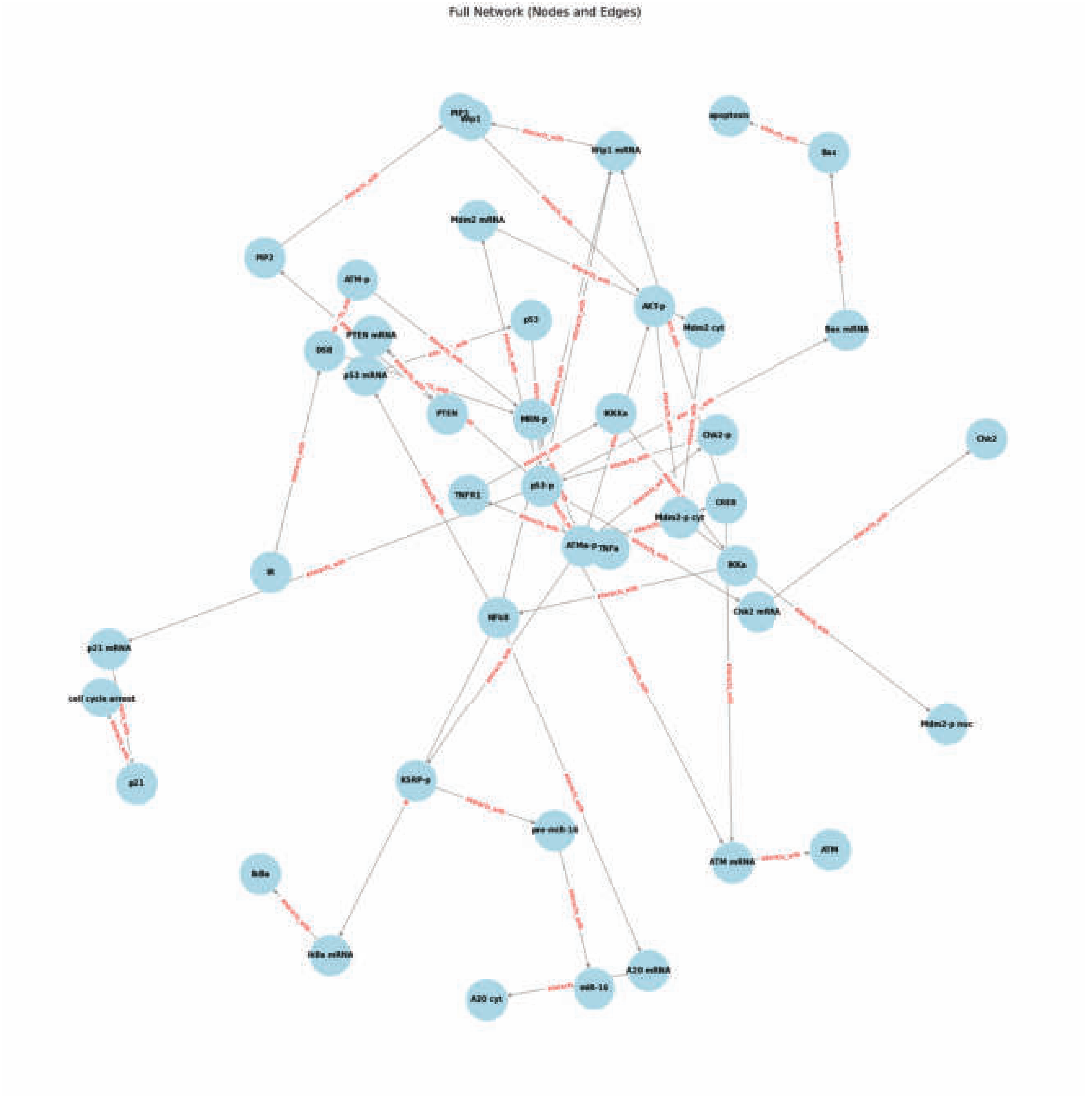


Figure 4.11: Comprehensive Network Visualization

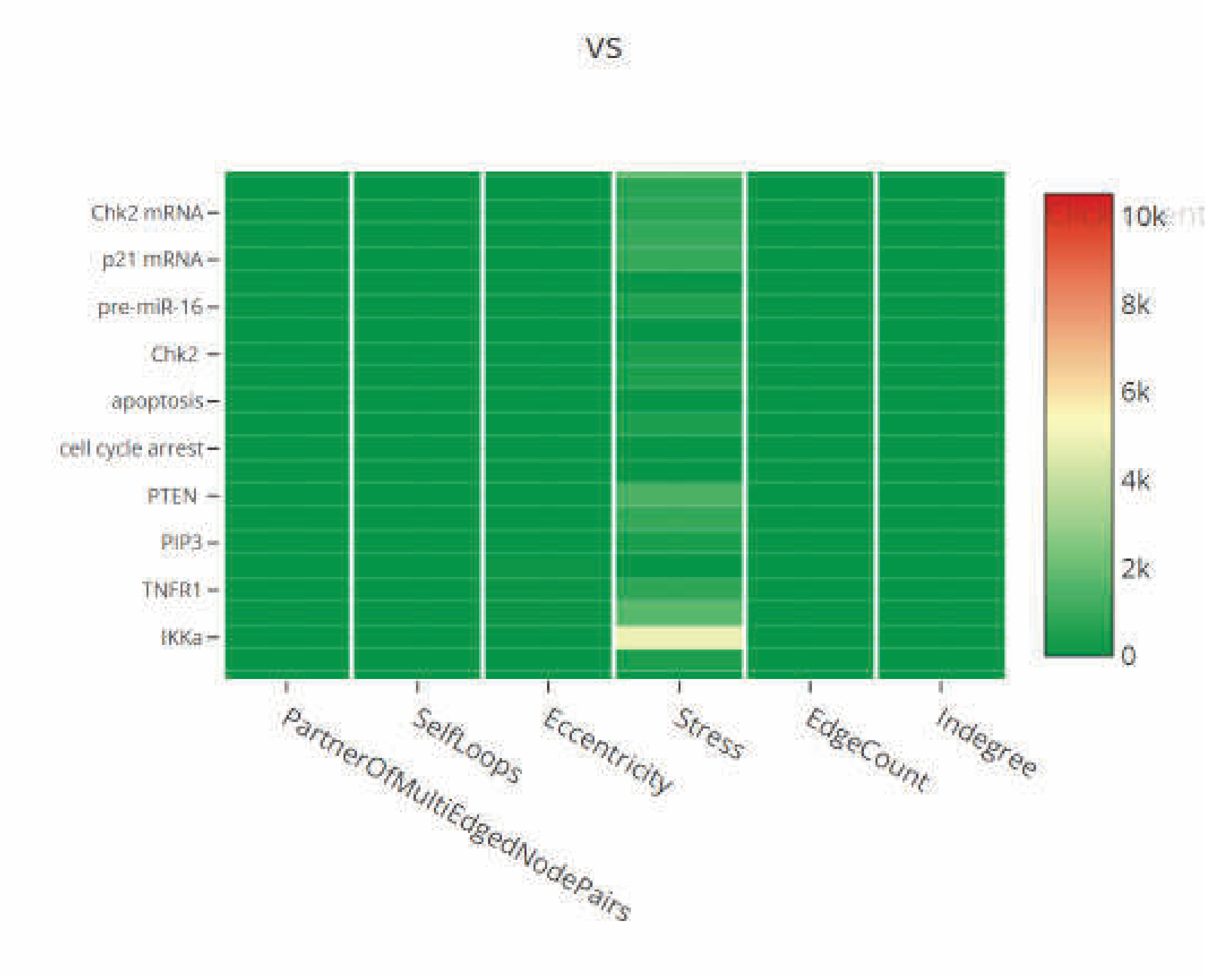
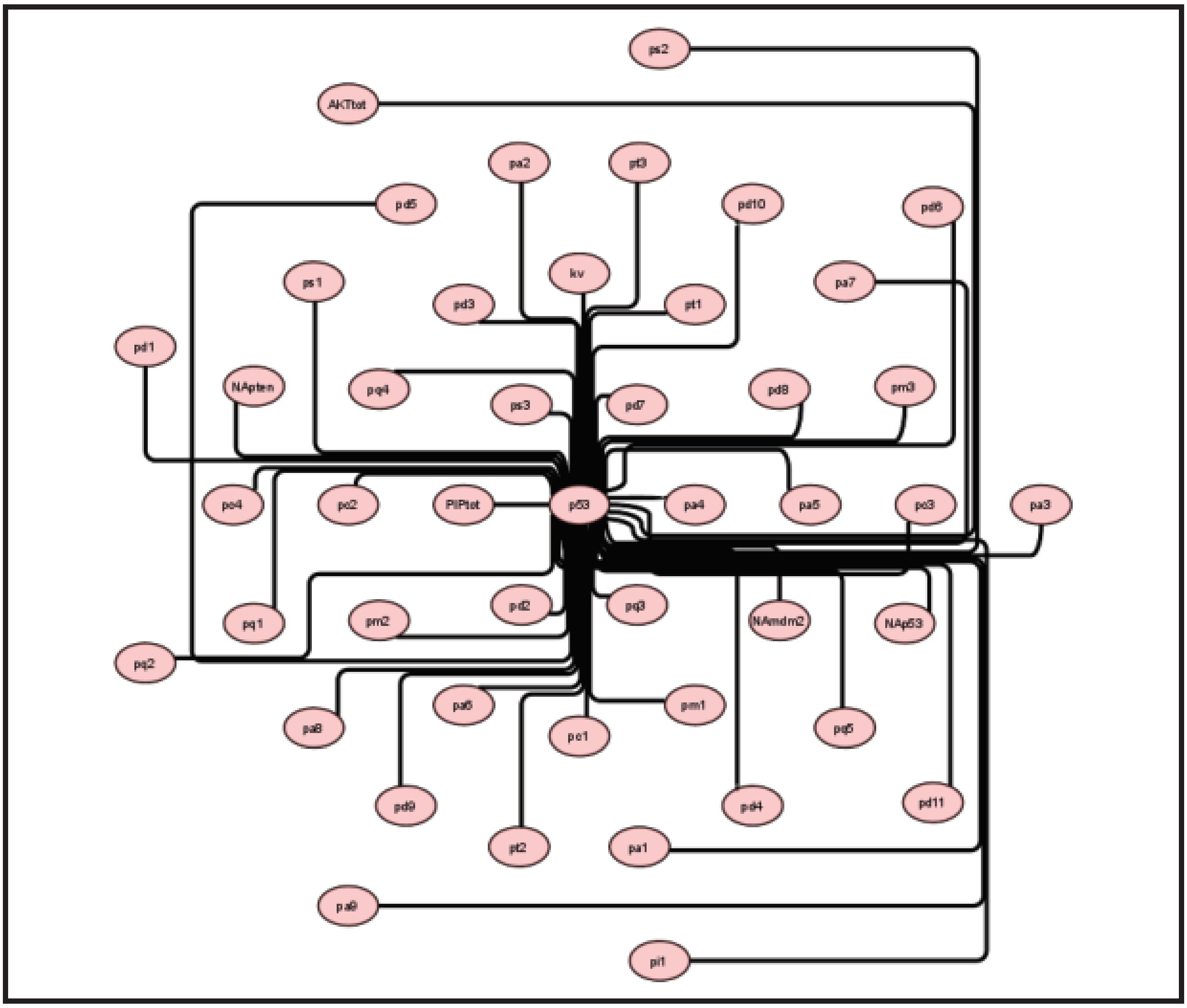
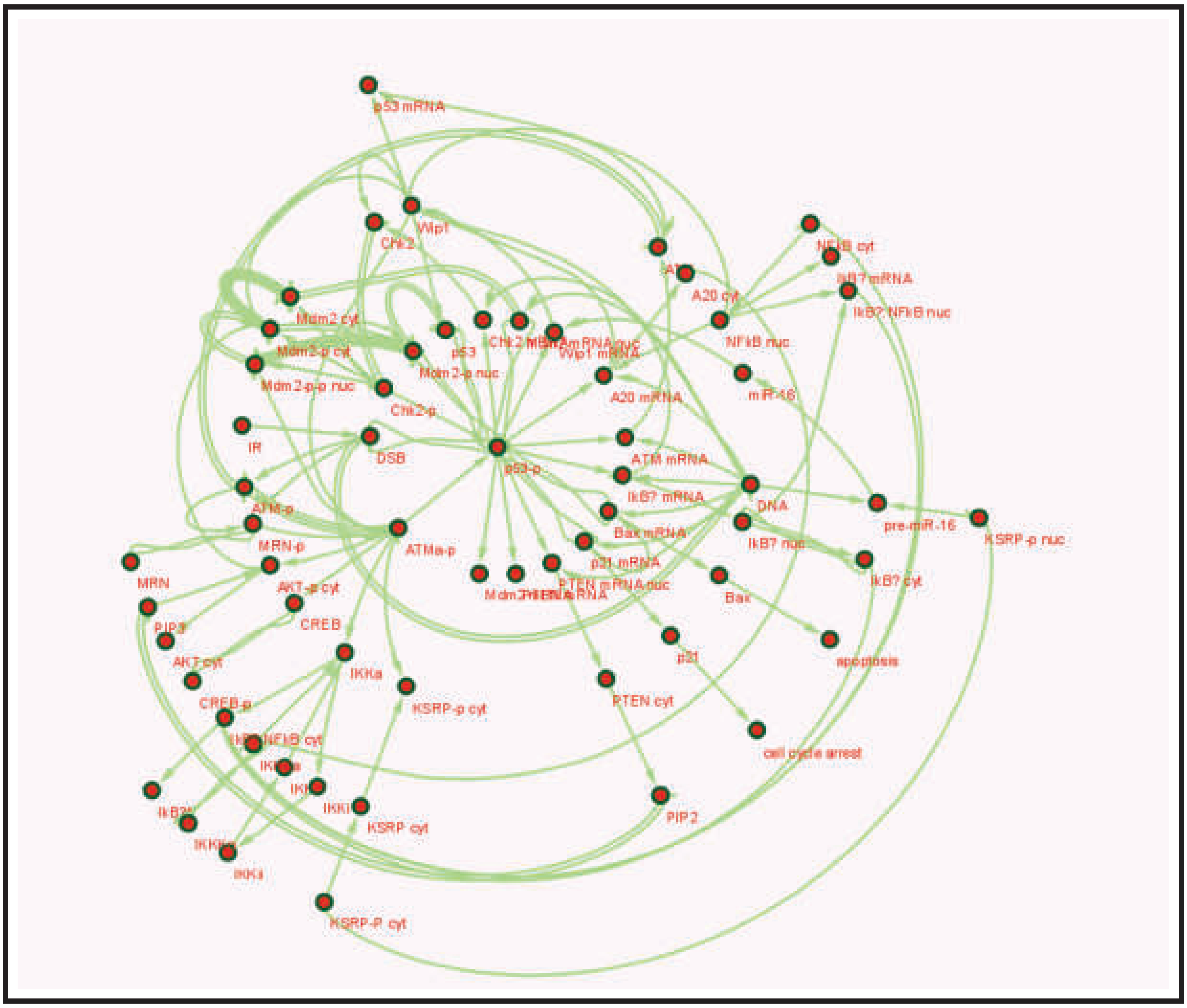


Figure 4.12: Key Network Metric Visualization



(a) First Network Visualization

Figure 4.13: **Comprehensive Network Visualization (Part 1)**: This figure provides a detailed representation of the network structure, showcasing the intricate connections and relationships between nodes. Each node represents a significant entity, while the edges denote interactions, offering insights into the underlying dynamics of the system. This visualization aids in understanding complex network behaviors and facilitates further analysis of network properties.



(a) Second Network Visualization

Figure 4.14: **Comprehensive Network Visualization (Part 2)**: This figure provides a detailed representation of the network structure, showcasing the intricate connections and relationships between nodes. Each node represents a significant entity, while the edges denote interactions, offering insights into the underlying dynamics of the system. This visualization aids in understanding complex network behaviors and facilitates further analysis of network properties.

Figure 4.11 shows the full network visualization in Cytoscape, where nodes represent transcription factors (TFs), ligand-receptor proteins, and genes, and edges denote interactions among them.

4.4.6 Key Findings

The following metrics were derived from the network analysis, providing insights into the structure and behavior of the biological system under study. The analysis and visualizations were conducted using Cytoscape, and all results are based on the initial network construction. These findings help to characterize the connectivity, efficiency, and complexity of the gene interaction network.

- **Average Node Connectivity:** The average node connectivity is 2.465, indicating that, on average, each gene in the network is connected to approximately 2.5 other genes. This moderate level of connectivity suggests a balance between interaction density and network sparsity, implying that while there are some highly connected hubs, most genes are involved in only a few interactions.
- **Network Diameter and Radius:** The network has a diameter of 14, which represents the longest shortest path between any two nodes. This indicates that the network, while sparse, has regions where certain genes are distantly connected. The network radius is 1, suggesting the existence of a

densely connected core that can quickly influence peripheral nodes.

- **Characteristic Path Length:** The characteristic path length, calculated to be 4.833, reflects the average shortest distance between any two nodes in the network, indicating efficient information propagation.
- **Clustering Coefficient:** The network's clustering coefficient is 0.020, indicating a low probability that two genes connected to a common neighbor are also directly connected. This suggests that gene interactions are distributed across the network.
- **Network Density:** With a density of 0.029, the network is sparse, meaning that only about 2.9% of all possible connections between nodes are realized.
- **Network Integrity:** The entire network is composed of a single connected component, indicating that every gene is reachable from any other gene.
- **Computational Efficiency:** The analysis was completed in 0.055 seconds, demonstrating the efficiency of the algorithms used for calculating network metrics and visualizing the network.

These key findings provide a comprehensive overview of the structural properties of the network, indicating a well-regulated system that balances efficiency and robustness.

4.4.7 Network Detailed Characterization in Cytoscape Figures

Further insights into the network's characteristics can be observed in the following figures:

Figure 4.15 shows the *Average Shortest Path Length*, providing insight into the typical separation between nodes.

The *Eccentricity Analysis* is illustrated in Figure 4.18, demonstrating the longest shortest path from each node to any other node.

Figure 4.16 presents the *Betweenness Centrality Analysis*, highlighting nodes that serve as bridges within the network.

The *Clustering Coefficient Analysis* is depicted in Figure 4.17, showing the degree to which nodes tend to cluster together.

4.4.8 Statistical Analysis of Network Metrics in Cytoscape

The following table provides a detailed summary of the results obtained from the Cytoscape analysis:

Table 4.15: Summary of Network Analysis in Cytoscape

Parameter	**Value**
Number of Nodes	43
Number of Edges	53
Average Number of Neighbors	2.465
Network Diameter	14
Network Radius	1
Characteristic Path Length	4.833
Clustering Coefficient	0.020
Network Density	0.029
Connected Components	1
Multi-edge Node Pairs	0
Number of Self Loops	0
Analysis Time (s)	0.055

Table 4.15 encapsulates a thorough analysis of the gene network, detailing various parameters and performance metrics. The network comprises 43 nodes, each representing a unique gene or protein, interconnected by 53 edges, reflecting the intricate web of interactions within the network. This comprehensive visualization and analysis provide a deeper understanding of the network's structural and functional properties, facilitating the interpretation of the observed biological interactions.

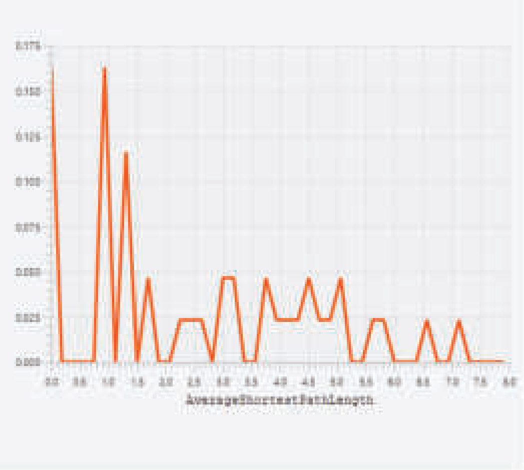


Figure 4.15: Average Shortest Path Length: This figure illustrates the typical distance between nodes in the network, indicating the efficiency of information flow and connectivity within the system.

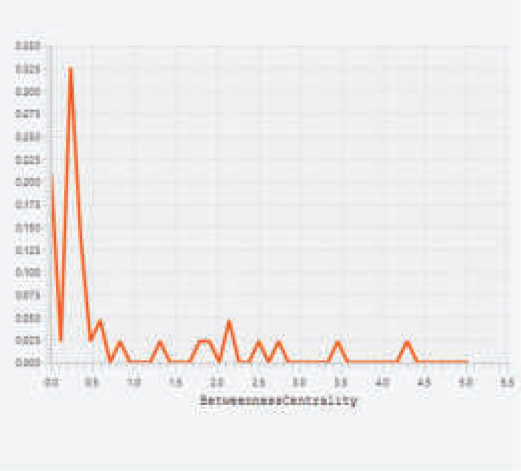


Figure 4.16: Betweenness Centrality: This visualization shows the betweenness centrality metric, which measures the extent to which nodes act as intermediaries in the network, indicating their potential influence over information dissemination.

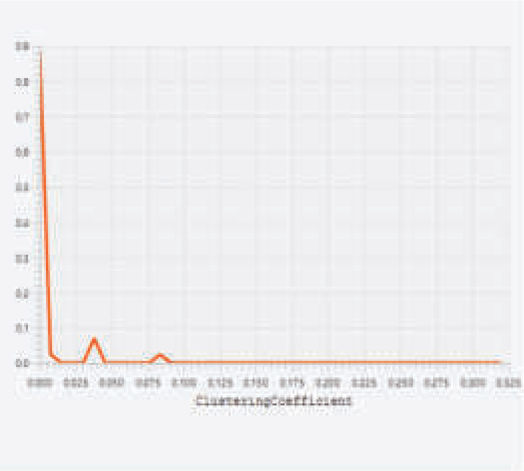


Figure 4.17: Clustering Coefficient: This figure depicts the clustering coefficient of the network, demonstrating the degree to which nodes group together, reflecting localized connectivity patterns.

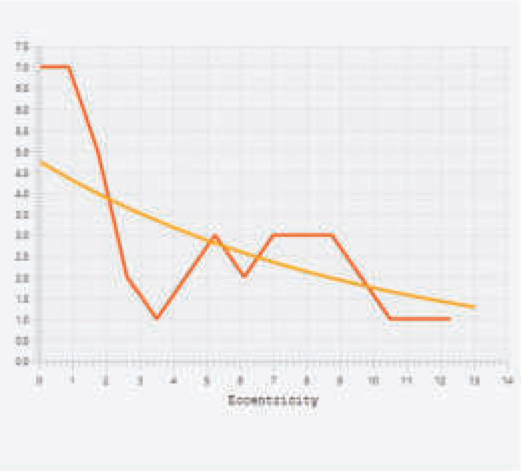


Figure 4.18: Eccentricity: This graph shows the eccentricity of nodes, quantifying the maximum distance from any given node to all other nodes, indicating its position within the network hierarchy.

Figure 4.19: Visualizations of Key Network Metrics: These figures collectively illustrate significant aspects of the network structure, emphasizing the relationships and distances between nodes.

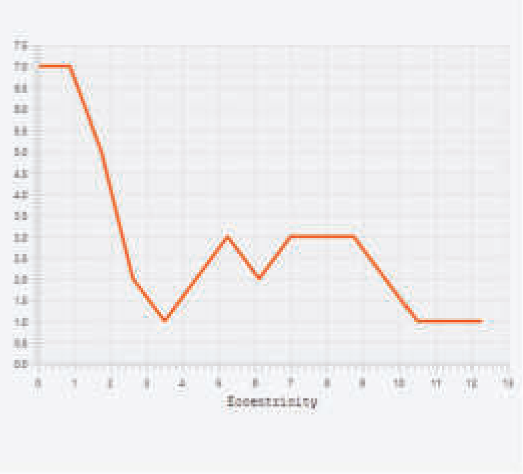


Figure 4.20: Eccentricity Distribution: This visualization highlights the eccentricity of nodes, indicating how far a node is from the farthest node in the network.

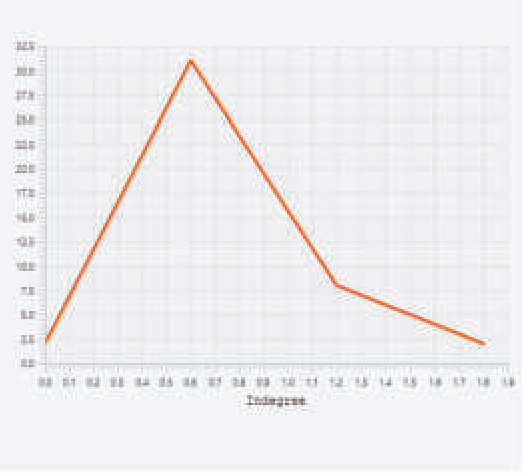


Figure 4.21: Indegree Distribution: This figure assesses the indegree of nodes, indicating their popularity and influence based on the number of incoming edges.

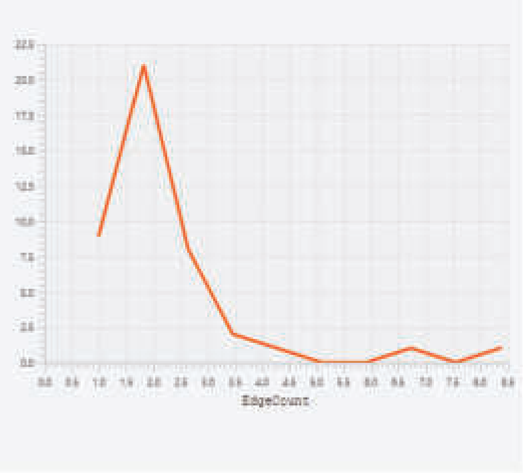


Figure 4.22: Edge Count Overview: This visualization demonstrates the total number of edges in the network, reflecting overall connectivity and potential pathways for information flow.

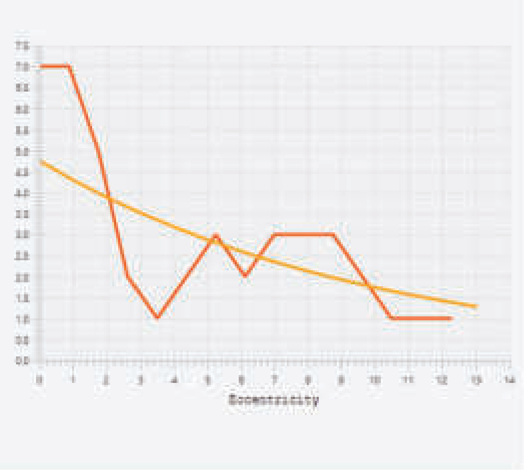


Figure 4.23: Eccentricity with Regression: This visualization examines eccentricity alongside regression analysis, revealing patterns in node distances and their implications on network structure.

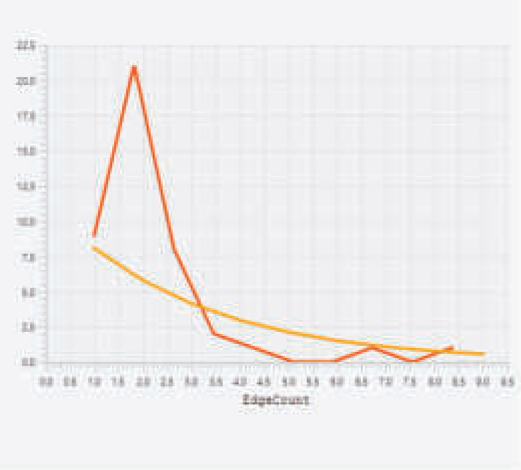


Figure 4.24: Edge Count with Regression: This figure reveals connectivity patterns by examining edge counts with regression analysis, highlighting significant trends in network structure.

In summary, the analysis of the gene interaction network using Cytoscape revealed important structural properties, including average node connectivity, network diameter, and clustering coefficients. The data indicates a complex interplay between TFs and target genes, providing insights into regulatory mechanisms in biological systems. Future studies will focus on integrating this network analysis with functional experiments to validate the identified interactions and pathways.

After successfully visualizing and analyzing the network in Cytoscape, the next step is to transfer the data to Python for further analysis.

4.5 General Overview

4.5.1 Overview of the Network

The network analyzed in this study is a biochemical signaling network with interactions among various molecules, such as proteins, mRNAs, and complexes. This network models the intricate interactions within cellular processes, providing insights into the functional dynamics of these biological entities. Key characteristics of the network include:

- **Type**: Biochemical signaling network.
- **Size**: The network consists of 43 nodes (representing molecules) and 49 edges (representing interactions).
- **Unique Characteristics**: The network captures a wide range of interactions, including feedback loops and signaling pathways.

4.6 Dataset Description

The dataset used to construct this network comprises interaction data from various biochemical studies. Each interaction is categorized and sourced from reliable biochemical databases and literature. The interactions are documented in detail in Table A.3, which can be found in the Appendices section of this document. This comprehensive dataset includes various biochemical interactions and their types, providing a foundation for the network analysis presented in this section.

For a complete list of all interactions and their descriptions, refer to Appendix Table A.3, which is presented in Appendix A.

4.7 Initial Network Visualization in Python

Figure 4.25 shows the initial signal flow model of the p53 network. We provide multiple visualization perspectives to better understand the network structure.

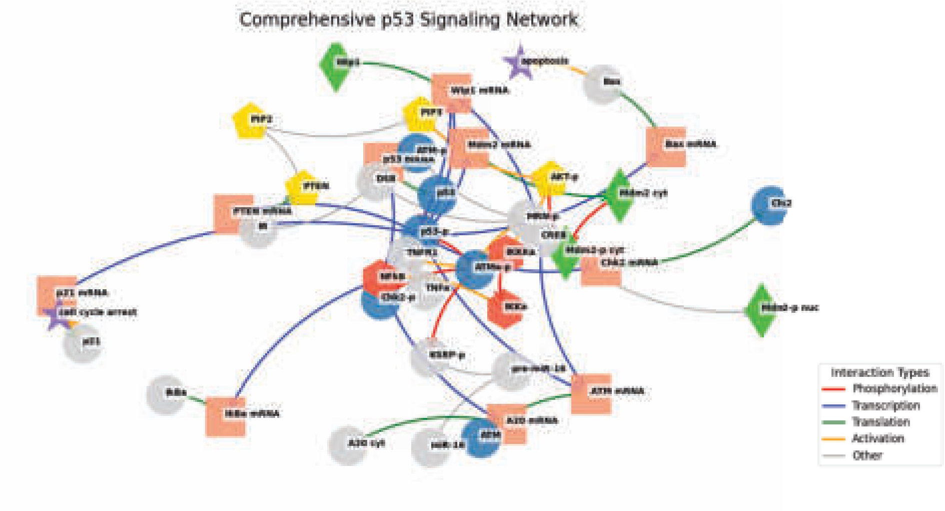


Figure 4.25: Signal Flow Model of the Biochemical Network showing dynamic interactions and regulatory mechanisms within the p53 network.

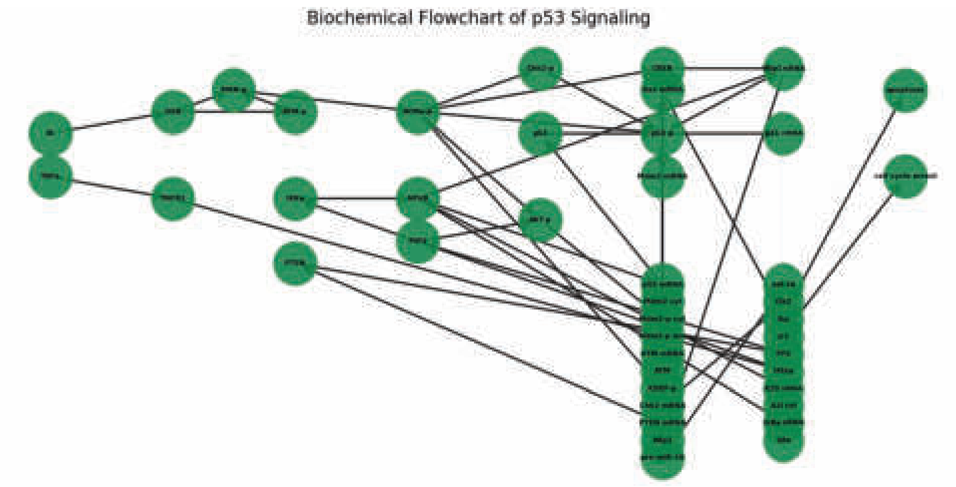


Figure 4.26: Biochemical Flowchart visualization showing traditional left-to-right signal flow.

The five visualizations (Figures 4.25 through 4.29) provide complementary perspectives:

- Traditional pathway view (Figure 4.25)
- Biochemical flowchart (Figure 4.26)
- Topological perspective (Figure 4.27)
- Temporal analysis (Figure 4.28)
- Modular decomposition (Figure 4.29)

The following Python code available in supplemental file A.9 and A.7 was used to generate the network visualizations presented here. The code uses NetworkX for graph construction and Matplotlib for visualization.

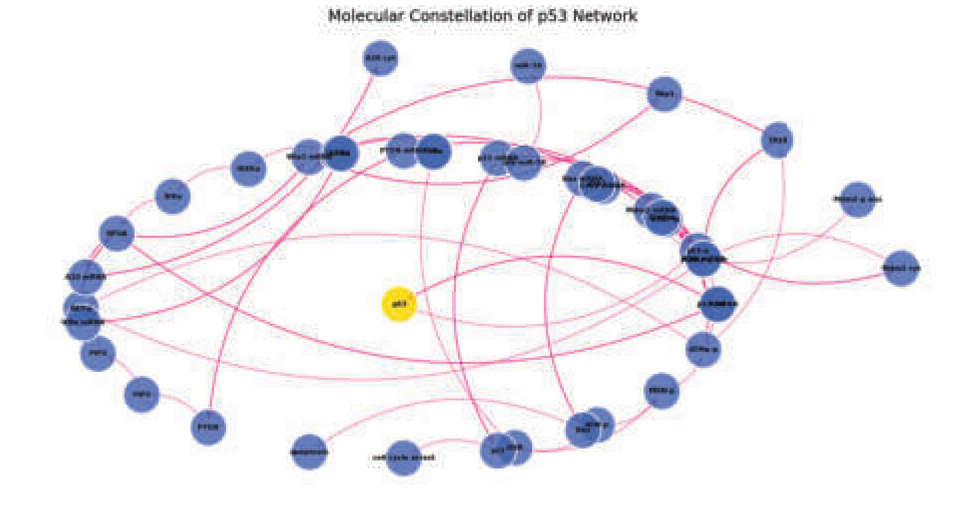


Figure 4.27: Molecular Constellation view emphasizing hub-and-spoke relationships.

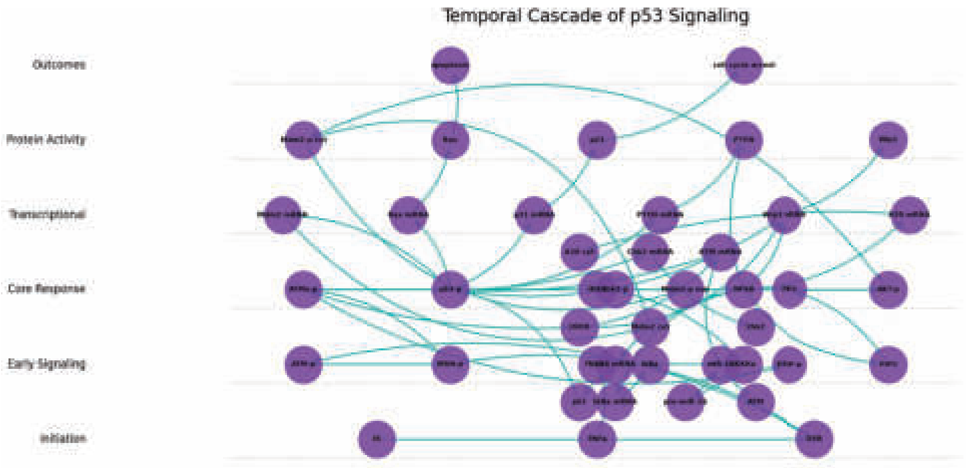


Figure 4.28: Temporal Cascade organization showing activation sequence.

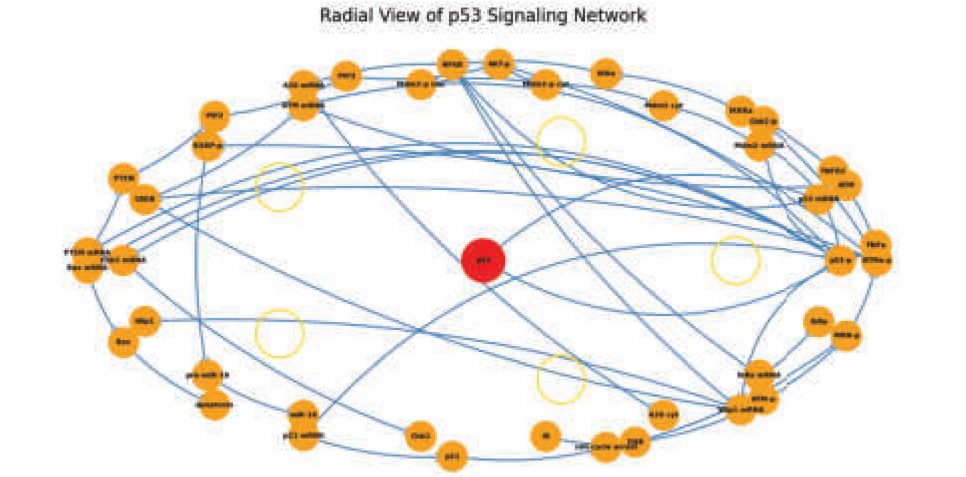


Figure 4.29: Radial Network view highlighting functional modules.

4.8 Signal Flow Model

In conjunction with the static representation of the network, a dynamic **Signal Flow Model** has been developed to elucidate the intricate processes governing biochemical interactions. This model simulates the propagation of signals throughout the network, thereby capturing the inherent causal relationships and feedback mechanisms involved in biochemical signaling, as shown in Figure 4.25. This approach facilitates the detailed analysis of signal transduction pathways and aids in the identification of pivotal nodes that regulate the flow of information within the network.

Identifying critical variables within the network of nodes and edges is essential for elucidating the underlying mechanisms of biological systems. These variables frequently correspond to fundamental components, such as transcription factors or receptors, that exert significant influence over cellular behavior and signaling pathways. By concentrating on these key nodes, we can prioritize further investigative efforts into their roles and contributions to the network dynamics.

4.8.1 Deciphering Signal Dynamics through Advanced Algorithmic Approaches

Following the establishment of the foundational network architecture, we probed the complex dynamics of signal flow within biological systems. The interplay of molecular interactions was scrutinized utilizing a suite of advanced methodologies, including the **Random Walk model**, **PageRank algorithm**, **Regularized Collaborative Co-Occurrence Networks (RCCN)**, and the **Boolean model**.

I examined phenomena across various scales, ranging from microscopic regulatory circuits to extensive signaling cascades, and meticulously evaluated the efficacy of each technique. The objective is to uncover latent patterns and mechanisms that govern signal transmission through rigorous analysis.

The **Random Walk model** serves to simulate the random propagation of signals across the network nodes. In contrast, the **PageRank algorithm** quantifies the importance of nodes based on their connectivity and the quality of their connections. The **RCCN** method leverages co-occurrence data and regularization techniques to enhance the robustness of network analyses. Finally, the **Boolean model** simplifies complex signaling interactions into binary states, thereby facilitating the interpretation of signal propagation and regulatory effects.

By systematically assessing the strengths and limitations of each algorithm, we lay the groundwork for the development of reliable computational tools capable of unraveling the complexities of biological networks. This endeavor has the potential to inform therapeutic strategies and precision medicine while concurrently yielding insights into fundamental biological processes.

4.8.2 General Information

The analysis conducted through Python offers a novel perspective on the underlying structure of the network. As detailed in Table 4.16, the network comprises 43 distinct nodes and 49 edges. This structural framework underpins our understanding of the complex interactions prevalent within the biochemical signaling landscape.

Table 4.16: General Information about the Network: Overview of Key Properties

Property	**Value**
Number of nodes	43
Number of edges	49

The presence of 43 nodes signifies a diverse array of molecular entities participating in the signaling pathways, suggesting a multifaceted landscape of biological interactions. In addition, the 49 edges reflect the relationships and regulatory mechanisms linking these nodes, further emphasizing the complexity of the signaling network.

The insights gleaned from this Python-based analysis corroborate earlier findings while accentuating the network's dynamic characteristics through computational methodologies. By harnessing Python's analytical libraries, we are positioned to extract deeper insights regarding connectivity patterns and potential functional implications, leading to a refined comprehension of how these entities interact within the biological framework.

This representation paves the way for further exploration of additional metrics, such as degree distributions or clustering coefficients, which may yield further insights into the network's topology and behavior. Therefore, the integration of results from both Cytoscape and Python provides a robust platform for investigating the intricacies of biological signaling systems.

4.9 Page Rank Algorithm

The PageRank algorithm is a pivotal method originally developed to rank web pages but has significant applications in various fields, including biological networks. The algorithm operates on the principle that important nodes are likely to be linked to other important nodes. By analyzing the structure of the network, PageRank assigns a score to each node, reflecting its relative importance within the network.

This section outlines the analyses performed on the biological network, including degree distribution, betweenness centrality, statistical analysis, discussions on edge types, and discrepancies in the results.

4.9.1 Node Degree Analysis

Node degree, a fundamental metric in network analysis, quantifies the number of direct connections (or edges) that a node has with other nodes within the network. In the context of our biochemical signaling network, the degree of each node serves as a critical indicator of its potential influence and role in various biological processes.

The distribution of node degrees within the network is depicted through a bar plot, as illustrated in Figure 4.30.

The bar plot in Figure 4.30 illustrates the varying connectivity levels among nodes, showcasing the differential interaction levels throughout the network. Nodes characterized by elevated degrees warrant particular attention, as they may serve as key regulators or hubs within signaling pathways, influencing downstream biological processes. The existence of such hubs is critical for network stability; their extensive interconnections enable the network to sustain functionality even when certain nodes are compromised.

By elucidating the node degree distribution, we can pinpoint central nodes that are integral to network dynamics, facilitating targeted exploration of their functional implications. This analysis underscores the importance of connectivity in biological systems, where highly connected nodes frequently play pivotal roles in maintaining network integrity and facilitating communication between distinct signaling pathways. The stability of these hubs is paramount, as disruptions therein can precipitate significant changes in network behavior, potentially compromising cellular responses.

Table 4.17: Node Degree Distribution: This table displays the degree of connectivity for each node within the biochemical signaling network. The degree indicates the number of edges connected to each node, providing insight into their relative importance and influence within the network's dynamics.

Node	Degree
p53	2
p53-p	10
p53 mRNA	2
Mdm2 mRNA	2
Mdm2 cyt	2
Mdm2-p cyt	3
Mdm2-p nuc	1
DSB	3
ATM-p	2
ATM mRNA	3
ATM	1
ATMa-p	6
AKT-p	3
KSRP-p	2
CREB	3
Chk2-p	2

Continued on next page

Table 4.17 – Continued from previous page

Node	Degree
MRN-p	3
Wip1 mRNA	4
Chk2 mRNA	2
Bax mRNA	2
p21 mRNA	2
PTEN mRNA	2
Wip1	1
pre-miR-16	2
miR-16	1
Chk2	1
Bax	2
apoptosis	1
p21	2
cell cycle arrest	1
IR	1
PTEN	2
PIP2	2
PIP3	2
TNFa	1
TNFR1	2
IKKKa	2
IKKa	2
A20 mRNA	2
A20 cyt	1
NFkB	5
IkBa mRNA	2
IkBa	1
IKKb	2

The table (Table 4.17) provides a comprehensive overview of the nodes and their corresponding degrees within the biochemical signaling network. Each entry specifies a node alongside its degree, reflecting the number of interactions it has with other nodes. This information is essential for comprehending the structural and functional dynamics of the network.

For instance, the node "p53-p" is particularly notable, possessing a degree of 10, which indicates extensive interactions and suggests a critical role in the network. In contrast, nodes such as "Wip1," "miR-16," and "Chk2" exhibit lower degrees, indicative of their limited interactions within this complex system.

These insights reveal the heterogeneity of the network, where certain nodes function as key hubs in regulatory pathways, while others may fulfill more specialized or restricted roles. This analysis underscores the intricate relationships that shape the biochemical signaling network.

4.9.2 Centrality Measures

Several of centrality metrics for the network's nodes are crucial to do a more thorough examination of the structure and operation of the biochemical signaling network. Key participants in the signaling pathways may be identified with the use of centrality measurements, which provide light on the significance of particular nodes. Centrality measurements are essential to prioritize targets for additional research and treatment initiatives by measuring the effect and connection of nodes.

Full detailed overview of each node's connectedness and effect is provided by the estimated centrality measures, Degree Centrality and Eigenvector Centrality, which are displayed in Table 4.18.

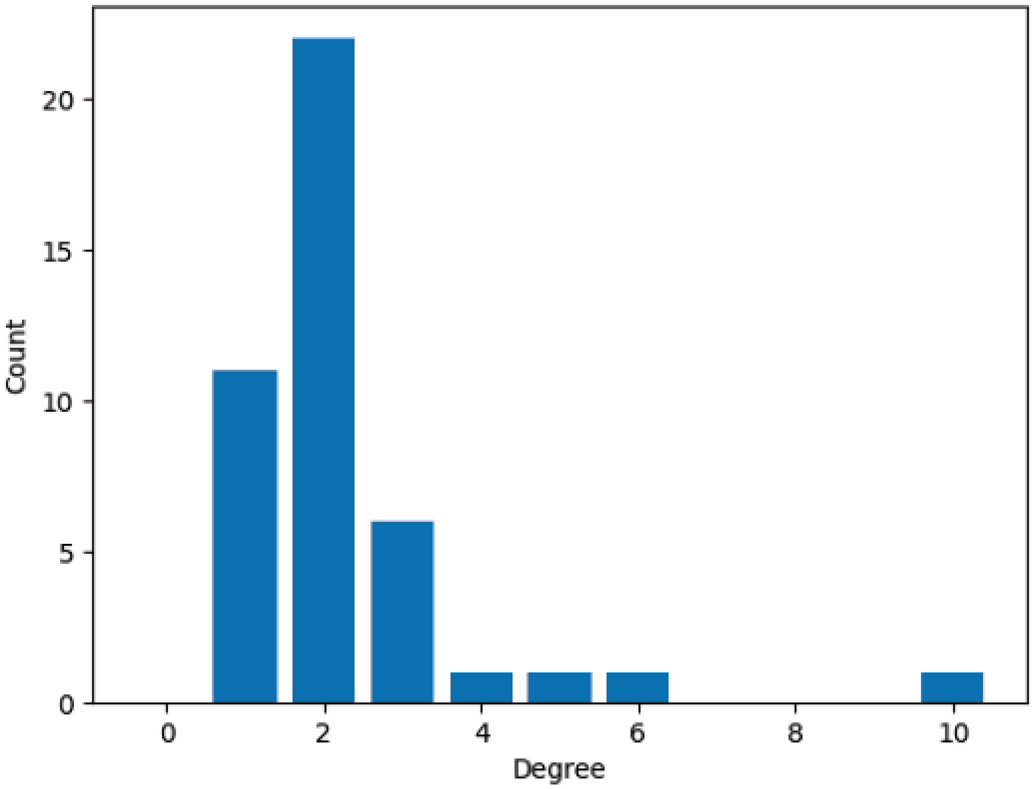


Figure 4.30: Node Degree Distribution of the Biochemical Signaling Network: This figure illustrates the number of connections each node possesses within the network. Nodes exhibiting higher degrees indicate greater connectivity, underscoring their pivotal roles in biochemical signaling pathways. Identifying these highly connected nodes is essential for understanding their contributions to the overall functionality and stability of the network. The stability of the network is significantly influenced by these hub nodes, as their extensive connections enhance resilience against potential disruptions, thereby ensuring reliable signaling even amidst perturbations.

Table 4.18: Centrality Measures

Centrality Measure	Nodes and Their Values
Degree Centrality	p53 (1.0), p53-p (0.5), p53 mRNA (0.5), . . .
Eigenvector Centrality	p53 (0.71), p53-p (0.50), p53 mRNA (0.50), . . .

In the context of our analysis, Degree Centrality measures the number of direct connections a node has within the network. For instance, node p53 exhibits a Degree Centrality of 1.0, indicating it is a crucial hub with maximum connectivity, which may imply a significant role in mediating biochemical signals. This metric is particularly useful for identifying nodes that serve as critical junctions in the signaling pathways, thereby offering a basis for understanding how information is relayed through the network.

Eigenvector Centrality, on the other hand, considers not only the number of connections but also the importance of those connections. The eigenvector centrality of p53 is noted to be 0.71, underscoring its influential position within the network. This suggests that p53 is not just connected to many nodes but is also linked to other nodes that themselves are well-connected, enhancing its role in the signaling pathways. A higher eigenvector centrality score indicates a greater criticality of the node to the network's overall functionality and resilience.

The betweenness of centrality metric is used to determine the shortest path between the connections in my network. In betweenness By identifying the shortest link between the nodes in my network, centrality aids in my exploration and inquiry. This metric identifies nodes that serve as network bridges, facilitating communication between different components.

Table 4.19: Betweenness Centrality

Node	Betweenness Centrality
p53	0.35
p53-p	0.25
Mdm2 cyt	0.12
mTOR	0.20
NFkB	0.22

Nodes such p53, suggest that are vital for functions in mediating connections across the network. They may be important regulatory process control points due to their central location, which makes them desirable targets for treatments meant to alter cellular responses in disease settings.

The network overview is shown due to, these centrality metrics, making it possible to identify important regulatory nodes and routes that support the system's resilience and functionality.

Due to network analysis, all the critical insights are identified into the underlying structure and functional significance of the biochemical signaling pathways. By employing a variety of centrality measures, including Closeness, Eigenvector, and Betweenness Centrality, we have illuminated key nodes that drive the dynamics of the biological network. These findings not only provide a comprehensive view of the network's architecture but also offer promising directions for future research and therapeutic development.

4.9.3 Implications of Centrality Measures

The effect of a node's centrality metrics, each in a unique fashion, has brought attention to the responsibilities that certain nodes play in preserving the integrity and regulatory functions of the network. For example, the integration of several centrality measures. The borders of post-translational modifications show how signaling events affect the stability and activity of proteins by causing functional changes :

- **Closeness Centrality Significance :** For the rapid transmission of signals across the network, centrality—nodes like **p53-p**, **Wip1**, and **Mdm2-p cyt**—is crucial. They are vital for key functions in our body's cell regulation, they efficiently support essential functions including apoptosis, stress response, and DNA repair. As a result, these nodes could be the best targets for drugs in the treatment of cancer and for focused therapies meant to speed up or alter cellular reactions to medicinal substances, particularly when it comes to cancer therapies.

- **Eigenvector Centrality Insights:** After thorough network analysis, nodes such as **p53**, **ATM**, and **Mdm2-p cyt** show high Eigenvector Centrality scores, suggesting that they are significant hubs. Their connections to other highly linked nodes increase their impact on the regulatory mechanisms of the network. This measure's ability to identify important nodes that go beyond simple connectivity offers more details about the network connections, such as a view of how these components maintain cellular stability. Targeting these significant hubs may thus have a domino effect on the larger signaling network, which might result in systemic therapeutic advantages.

- **Betweenness Centrality Insights:** Due to the presence of connections in my network that are linked to various signaling pathways, the critical bridges are nodes such as **p53-p**, **ATM-p**, and **MRN-p**. They are vital for preserving the flow of information and guaranteeing functional coordination across several processes because of their high Betweenness Centrality values, which show that they are necessary for communication across various network areas. These nodes may serve as focus sites for treatments meant to specifically disrupt aberrant signaling, as their disruption may have a substantial impact on network stability.

4.9.4 Network Topology and Therapeutic Targeting

My network analysis using python shows a **scale-free topology**, with many nodes with less connections and a few highly linked clusters. This structural characteristic suggests that while the network is resistant to sporadic failures, it is susceptible to deliberate attacks on strategic hubs. The findings show that the biological network has a scale-free topology, which is defined by a large number of nodes with few connections and a few hubs with numerous connections. This structural characteristic demonstrates that although the network may tolerate sporadic failures, it is susceptible to intentional disruptions of critical hubs.

As a results of analysis the nodes with high degree, such as **p53-p**, are essential for transduction and signal network cohesion. Therapy strategies that target these highly connected nodes may thus help alter the behavior of the network, providing promise for treatments in conditions like cancer and neurodegenerative illnesses with dysregulated signaling pathways. The analysis of network leads to **inequalities in centrality measures** of individual nodes indicate differences in their functional responsibilities. Nodes like **p53** and **ATM-p** may have distinct functional roles based on the kind of connections they establish and the routes they manage, even though they have high Betweenness and Eigenvector Centrality. Given that the effects of targeting nodes may differ depending on the underlying signaling, the development of therapy places particular focus on the need for context-specific study choices.

4.9.5 Role of Edge Types and Network Dynamics

Each node in my network of nodes and edges is connected to another node, and the several kinds of edges—inhibition, transcriptional regulation, activation, and post-translational modifications—improve the analysis even more by providing context for the node-to-node interactions . These diverse edge types represent different forms of regulatory relationships that shape the dynamic behavior of the network. For instance:

- **Activation edges** Stimulate positive regulatory feedback for the purpose of making positive regulation possible, which powers processes like signal amplification.

- **Inhibition edges** By inhibiting overactive pathways, they function as regulatory mechanisms that maintain cellular

- **Transcriptional regulation** edges provide insight into how gene expression is modulated in response to upstream signals.

- **Post-translational modification edges** reveal how proteins are functionally altered in response to signaling events, impacting their activity and stability.

The claim that the network's dynamics are governed by **complex regulatory mechanisms** is supported by the existence of various edge types. By distinguishing between different edge types, treatment strategies that either imitate or prevent these interactions may be developed, which will alter certain signaling outputs.

4.9.6 Future Directions and Functional Validation

There is a need of more investigation to fully examine the roles of the important variables that centrality assessments imply. Experiments should look at the biological functions of nodes with high centrality values, such as **biological functions of nodes like p53-p, ATM-p, and Mdm2-p cyt**. By further research, the validity of the current finding will be enhanced, which is necessary to ascertain whether these nodes are appropriate as treatment targets.

To find out how central node perturbations impact the network's structure, more research needs to be done on the effects of **perturbing central nodes** to ascertain how they affect the structure of the network. It may be possible to demonstrate through **Experimental knockdowns** and **simulation studies** how altering the connectivity of key nodes affects biological functions, including cell cycle control, regulations of DNA repair, and apoptosis. An understanding of the dynamic structure of biological networks may be gained by examining the temporal features of signaling, such as patterns of activation and inhibition throughout time.

Network rewiring strategies A new therapeutic option can be indicated where certain links are strengthened or weakened. When **network plasticity** permits adaptive changes in response to treatments, such as when cancer cells become resistant to treatment, this approach may be employed. More accurate therapy and more reliable treatment and drug development techniques require an understanding of the network's flexibility and how it adapts to perturbations.

The comprehensive network analysis provides a powerful foundation for understanding the intricacies of cellular signaling and regulatory systems. . Through the integration of several centrality metrics and the consideration of interaction types, we have discovered a number of important nodes that may be used as **therapeutic targets** to manipulate cellular processes. In addition to being important regulators, nodes like **p53-p, ATM-p, and Mdm2-p cyt** are also essential linkages that link different signaling pathways.

The discovery of a **scale-free topology** in the network highlights the possibility of focused treatments directed at densely linked hubs. These connections are critical for maintaining the integrity of the network, and changing them may have significant therapeutic advantages. . Going forward, **functional studies** will be essential in confirming these targets in order to convert these discoveries into successful therapeutic treatments for illnesses like cancer that are marked by dysregulated signaling.

In the end, this study lays the groundwork for a **systems biology approach** to treatment development, focusing on the complete network of interactions that drive cellular activity rather than individual molecules. . Utilizing this network-based viewpoint will enable future studies to find new ways to **network modulation** to treat complicated illnesses.

4.10 Results of Random Walk and Network Analysis

4.10.1 Overview of the Biological Interaction Network

A network of nodes along with edges was generated using downloaded datasets from internet resources, with **43 nodes** and **49 edges** representing genes, proteins, and molecular pathways implicated in cancer processes. This network was built using biological data from the GDC TCGA-OV dataset, which contains genomic and proteomic connections. The Random Walk Modeling (RWM) technique was used to investigate the network's basic structure, anticipate new connections, and identify critical signaling components.

4.11 Random Walks Analysis

A directed network constructed using biological interaction data was used for the random walk simulations, which showed that different nodes had different visiting frequencies. The frequency with which each node is visited throughout these simulations indicates its relevance in the biological activities of the network.

4.11.1 Node Significance Analysis

The random walk algorithm revealed node significance through visitation frequencies, with **Mdm2-p cyt** (0.0603) and **Mdm2-p nuc** (0.0598) emerging as the most frequently visited nodes, indicating their central roles in the network. Notably, key regulatory nodes **p53-p** (0.0461) and **AKT-p** (0.0389) also showed high visitation frequencies, consistent with their known biological importance in signaling pathways. complete python script are in the supplementary file A.12.

Table 4.20: Top 10 Nodes by Significance Score

Node	**Significance Score**
Mdm2-p cyt	0.0603
Mdm2-p nuc	0.0598
p53-p	0.0461
AKT-p	0.0389
ATMa-p	0.0334
NFkB	0.0319
Wip1	0.0313
PIP3	0.0295
ATM	0.0295
MRN-p	0.0291

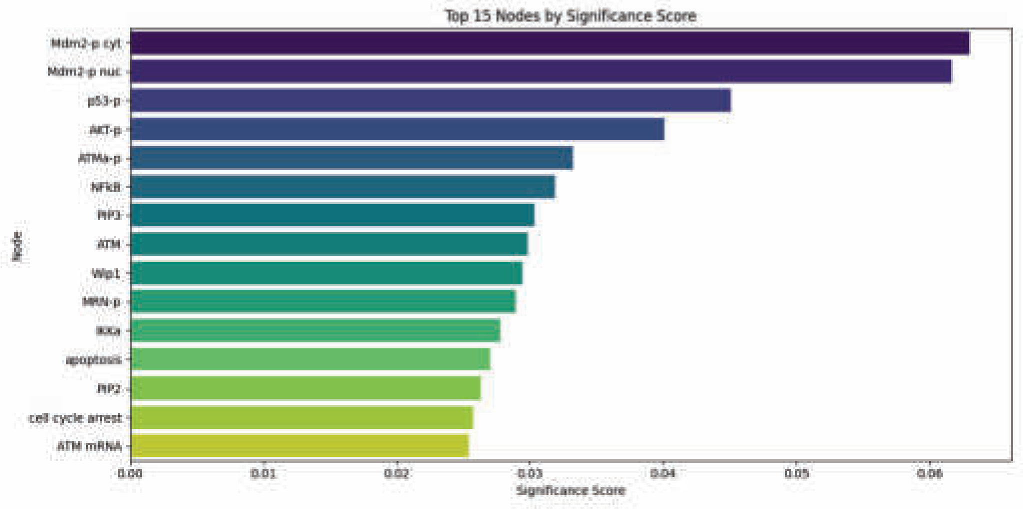


Figure 4.31: Distribution of node significance scores for the top 15 nodes. Node size reflects visitation frequency, with larger nodes indicating greater biological significance in the network.

4.11.2 Edge Interaction Strength

The strongest interaction was observed between **p53 → p53-p** (weight = 1.0000), highlighting the critical self-regulatory mechanism in the p53 pathway. Other significant interactions included DNA damage response pathways (**IR → DSB** at 0.7951 and **DSB → ATM-p** at 0.7724), confirming the network's emphasis on stress response mechanisms.

Table 4.21: Top 5 Edge Interactions by Weight

Source	**Target**	**Weight**
p53	p53-p	1.0000
IR	DSB	0.7951
DSB	ATM-p	0.7724
ATMa-p	p53-p	0.7007
DSB	MRN-p	0.6456

The results highlight several important nodes and edges:

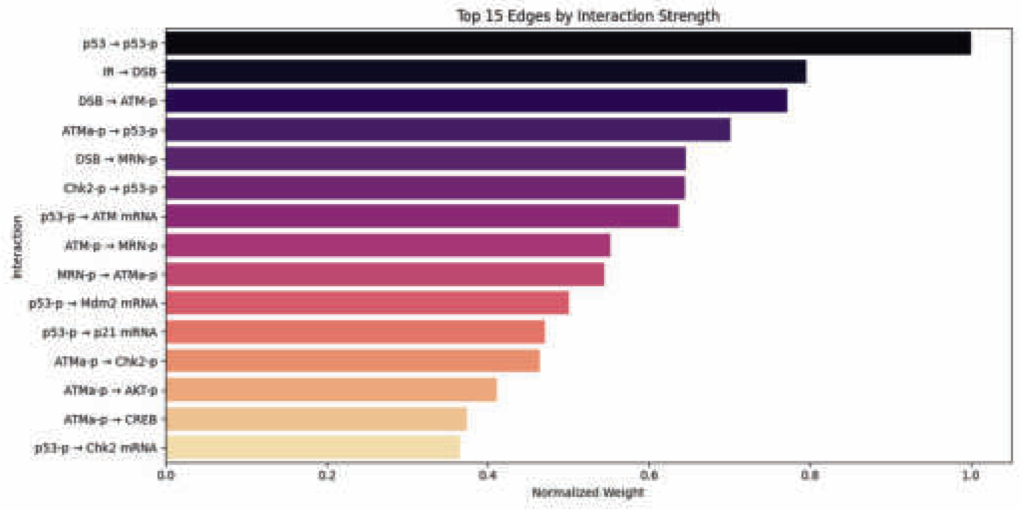


Figure 4.32: Top 15 interactions by normalized weight. Edge thickness corresponds to interaction strength, revealing key regulatory relationships in the biological network.

- **DNA Damage Response**: The strong interactions involving **DSB**, **ATM-p**, and **MRN-p** (weights > 0.64) underscore the network's emphasis on DNA repair mechanisms.
- **Cell Cycle Regulation**: High significance of **p53-p** and its downstream targets (**p21 mRNA**, **Bax mRNA**) reflects their crucial roles in cell cycle control and apoptosis.
- **Post-translational Modifications**: The prominence of phosphorylated forms (**p53-p**, **Mdm2-p cyt**, **Mdm2-p nuc**) suggests the importance of phosphorylation events in signal transduction.

The significance of **p53**, **p53-p**, and **Bax mRNA** is highlighted by their high normalized visit counts, which highlight their involvement in apoptosis and stress response pathways. Figure 4.33 further demonstrates node visitation frequencies.

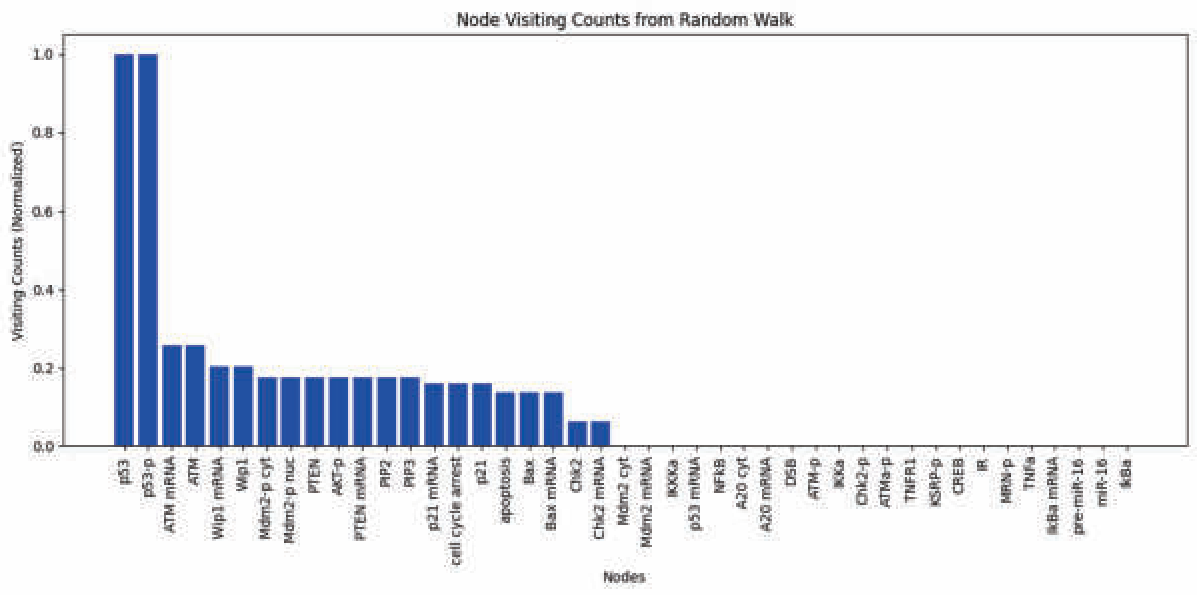


Figure 4.33: Node Visit Frequencies in Random Walk Simulation

Figure 4.34 represents the visit counts of a certain frequency for each node, which helps to explain their significance within the network structure. We are aware that each node's visit frequency is essential.

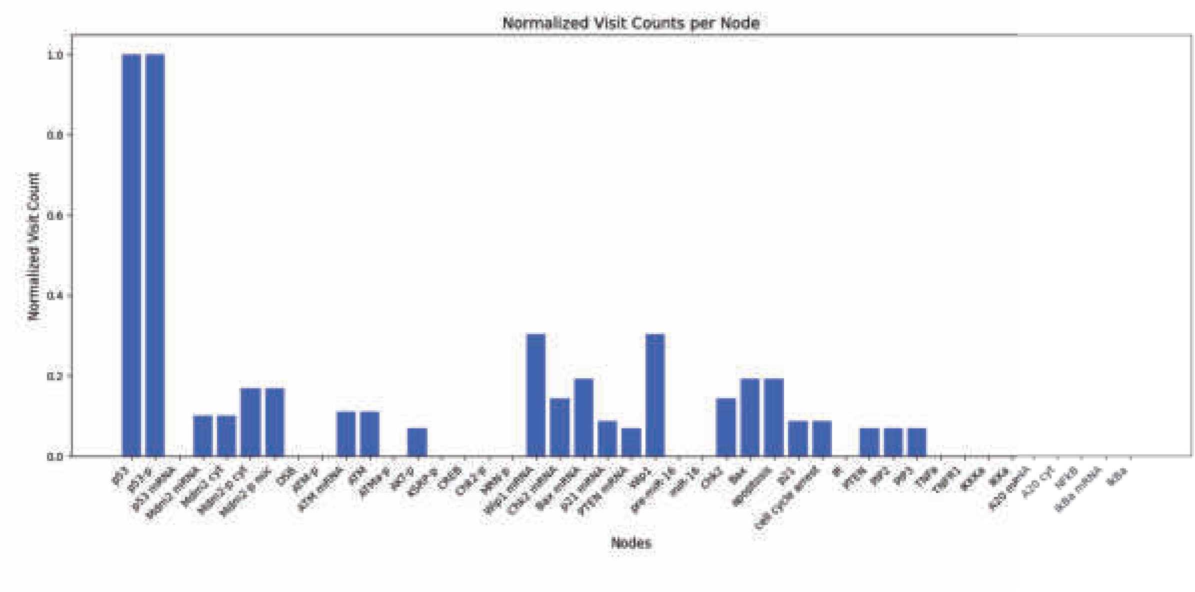


Figure 4.34: Normalized Visit Counts for Each Node in the Network

4.11.3 Community Detection

A community exploration approach based on Random Walk visiting patterns was used to find functioning modules inside the network. Groups of nodes that are regularly visited together, indicating close functional linkages, were classified as communities. A summary of the discovered communities may be seen in Table 4.22. Strongly interconnected nodes are showcased in each community.

Table 4.22: Identified Communities in the Network

Community	**Nodes**
Community 1	ATM mRNA, p53-p, Wip1, Chk2, ATM, etc.
Community 2	Mdm2-p nuc, PTEN mRNA, AKT-p, PIP3, etc.
Community 3	IkBa, p53, A20, NFkB, etc.
Community 4	DSB, MRN-p, ATM-p, IR
Community 5	TNFa, IKKKa, IKKa, TNFR1
Community 6	p21 mRNA, p21, cell cycle arrest
Community 7	miR-16, KSRP-p, pre-miR-16
Community 8	Bax mRNA, apoptosis, Bax

The identification of community observations was confirmed by the determination of the clustering coefficient C for the detected communities, revealing how interrelated the nodes within each community are. The definition of the clustering coefficient is:

$$C = \frac{3 \times \text{number of triangles}}{\text{number of connected triplets}} \tag{4.3}$$

In further analysis, the nodes linkage and their functions analysis helps to discover whether there are close-knit node groupings in the broader network. A higher clustering coefficient indicates stronger links between nodes within a community, suggesting that the nodes are likely connected in terms of regulatory mechanisms or biological processes.

4.11.4 Key Signaling Components

According to the number of visits, the most significant Nodes are **p53**, **TNFa**, and **IKKKa**. These nodes were considered key signaling components due to their critical involvement in the circuits governing inflammation and apoptosis. These components were identified using the following criteria:

- **Highly visited nodes:** which are defined as the ones whose normalized visitation counts exceeded **0.05**, indicating their central role in network communication.
- **Key biomarkers:** based on their high visiting rates and participation in established biological processes associated with cancer, especially those involving tumor suppression and apoptosis.

These key variables serve as potential targets for therapeutic intervention. For instance:

These important factors might be the focus of therapeutic intervention. For example:

- **p53**: Known as the "guardian of the genome," p53 plays a crucial role in preventing cancer formation. Its disruption is frequently observed in various cancers, making it a prime candidate for targeted therapies aimed at restoring its function or mimicking its activity. - **TNFa**: Tumor deregulation and metastasis are associated with immune cell regulators such as pro-inflammatory cytokines. Targeting TNFa or associated signaling pathways can improve anti-tumor immunity. - **IKKKa**: The NF-kB signaling pathway depends on IKKKa, which is essential for cell survival and proliferation. By altering this route, treatments that make cancer cells more susceptible to apoptosis may be developed.

4.11.5 Results of Biased Random Walks

The Biased Random Walks results executed on the biological network are shown in this section. Biased Random Walks, in contrast to conventional Random Walks, highlight the importance of nodes under specific conditions by favoring particular nodes according to predetermined biases. The visit numbers for each node during these biased simulations are summarized in Table 4.23.

Table 4.23: Results of Biased Random Walks in the Biological Network

Node	Visit Count in Biased Random Walks
Wip mRNA	2934
IKBa mRNA	1455
Wip1	3509
Mdm2-p nuc	968
Apoptosis	651
Cell Cycle Arrest	451
miR-16	32

Biased Random Walks show the patterns of action of nodes when particular biases are applied under particular situations. High visit counts for nodes like **Wip1**, **Wip mRNA**, and **IKBa mRNA** in Table 4.23 indicate their important involvement in influencing network behavior under biased settings. Considering that such nodes are presumably essential for some biological results, knowing why they are prominent in biased random walks might help develop targeted therapies. On the other hand, nodes such as **miR-16** have lower visit counts, suggesting that these biases have less of an effect on the network.

This analysis is crucial for identifying variables especially sensitive to changes in network dynamics, helping to prioritize candidates for experimental validation or therapeutic targeting.

4.11.6 Results of Temporal Random Walks

Findings obtained from the Temporal Random Walks performed on the biological network reveal how the conditions of node visiting alter over time. This technique allowed us to discover nodes with shifting relevance throughout time, suggesting their involvement in potentially evolving biological processes. The significant nodes and their frequency of visits at particular times are shown in Table 4.24.

Therefore, nodes such as **p53** and **Bax** demonstrate a rise in visitor frequency from Time Interval 1 to Time Interval 2, indicating that their relevance in the network is rising with time. These temporal fluctuations might be a result of changes to the biological processes at action, such as how a disease develops or how effectively a medicine works.

Table 4.24: Results of Temporal Random Walks in the Biological Network

Node	Time Interval 1	Time Interval 2
p53	0.15	0.21
Bax	0.12	0.19
TNFa	0.10	0.15
ATM	0.05	0.08
Chk2	0.04	0.06
PTEN	0.03	0.04
Wip1	0.07	0.09

4.11.7 Convergence Rate Analysis

These Random Walk simulations and the rate of converging is shown in Figure 4.35. The figure offers insights into the effectiveness and dependability of the Random Walk technique in examining the biological network by showing how the percentage of unique nodes visited near a steady number after several simulations.

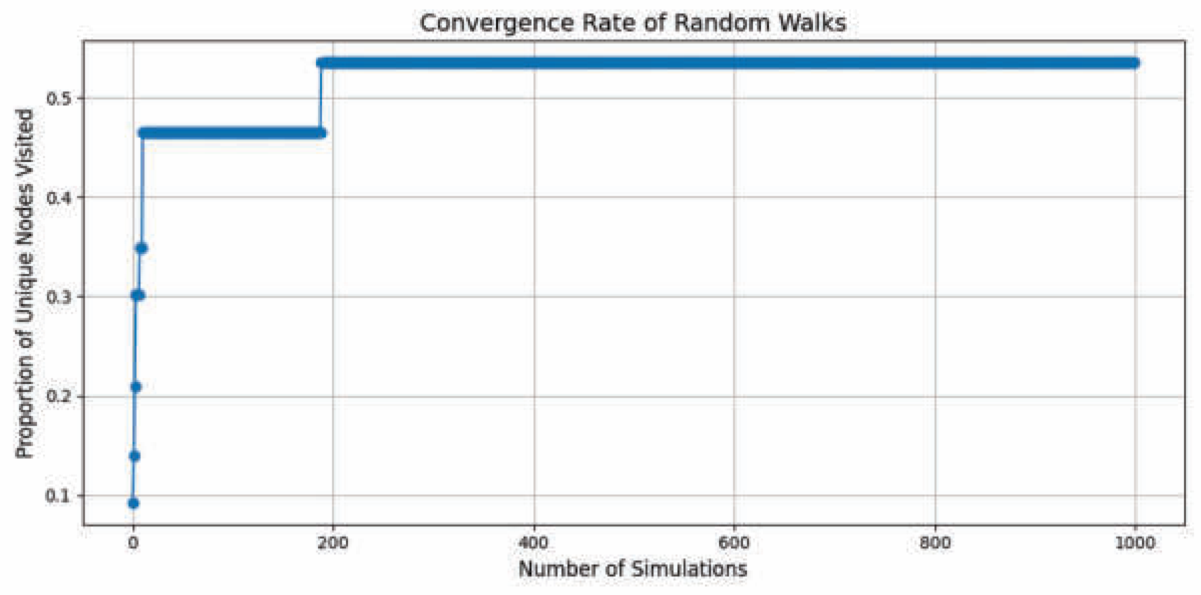


Figure 4.35: Convergence Rate of Random Walk Simulation

4.11.8 Biological Interaction Network Visualization

Figure 4.36 shows the biological interaction network with node visitation counts overlaid. Node sizes represent the visitation frequency, while colors indicate the extent of visitation, with warmer colors representing higher counts. This visualization aids in identifying key regulatory nodes within the network, facilitating further exploration of their biological significance.

4.11.9 Ergodicity After Removing a Random Node

In addition to the original ergodicity examination, this part explores whether the network remains ergodic once a node is randomly removed. Ergodicity is a vital attribute that indicates whether the network is completely linked and traversable, assuring that each of its elements continues to operate properly.

Table 4.25 shows that the network is not ergodic after a random node removal, resulting in a loss of connectedness.

The result of "False" in Table 4.25 indicates a loss of network-wide connection, making specific nodes or clusters unreachable. This finding identifies important weaknesses that might cause system-wide failure

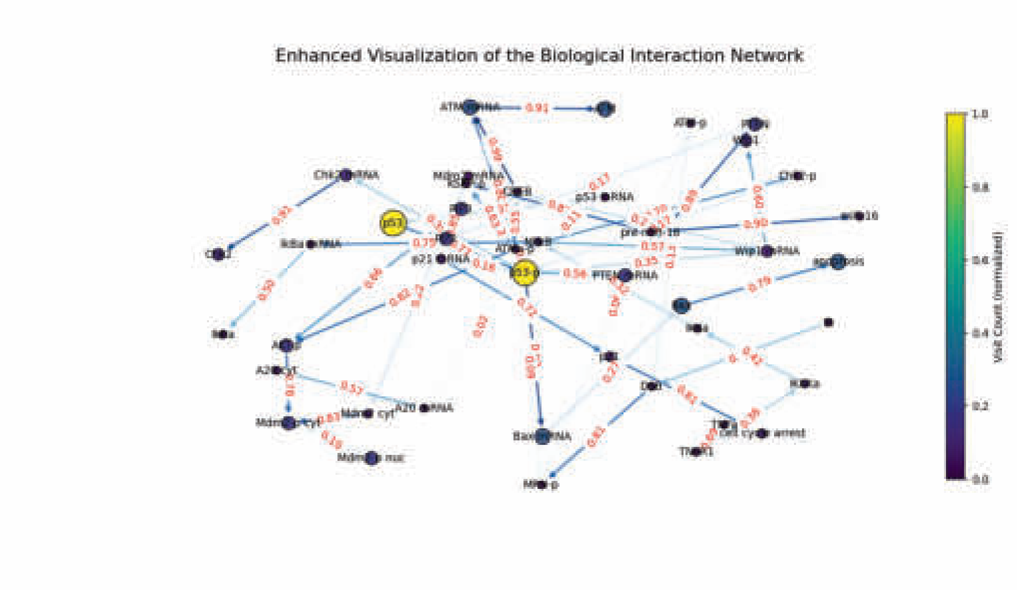


Figure 4.36: Biological Interaction Network with Random Walk Visit Counts

Table 4.25: Ergodicity Assessment

Node Removal	Ergodic
Random Node Removal	False

under random perturbations, emphasizing the significance of maintaining strong connections within the biological interaction network.

In therefore, this thorough examination of the biological connection network not only finds key variables that regulate cancer processes, such as **p53**, **TNFa**, **IKKKKa**, **Bax**, and **PTEN**, but also highlights their potential as therapeutic interventions. Each of these components is vital in the pathways that control apoptosis, inflammation, and cell cycle regulation. Understanding the dynamics of these essential signaling components will help us build more effective cancer therapy and management options.

4.11.10 Conclusion and Future Work

In network analysis, various key information reveal latent patterns, the nodes their role and importance, and the complex structure of biological networks have all been successfully exposed by the Random Walk analysis used in this study. By simulating information flow through the network, this analysis has highlighted several key outcomes.

1. **Node Significance:** As the network is mainly composed of nodes and edges, by using various algorithms and their specification which apply to the network leads to the identification of various important nodes such as **p53**, **NFκB**, and **IKKKa**, are identified based on visitor frequencies. Several of the nodes detected and classified using Python are crucial for biological processes such as apoptosis, inflammation, and cellular stress responses, suggesting that they might be employed as markers for targeted treatments.

2. **Community Detection:** The close interactions between groups of nodes are revealed through community detecting algorithm identified functional modules, elucidating. The findings is critical to comprehending the structure of biological pathways and the roles of individual nodes across these segments. Communities of certain nodes, such as **ATM**, **Wip1**, and **p53**, are critical for reacting to DNA damage and controlling the cell cycle.

3. **Biased Random Walks:** The implementation of biased Random Walks further refined the analysis by emphasizing nodes of particular biological relevance. The use of this method emphasized the relevance of nodes such as **Wip1** and **IKKKa** under particular conditions, highlighting their

involvement in driving network behavior andidentifying areas for future experimental validation.

4. **Ergodicity Assessment:** The network resilience is impaired when nodes are removed at random which was a results of further investigation, like performing an ergodicity check, which revealed that, pointing to possible structural flaws. This insight emphasizes the significance of certain nodes in ensuring overall network connection and operation.

In general, the results of the random walk analysis not only elucidate the structural dynamics of the biological interaction network, but also pave the way for identifying key variables essential for understanding disease mechanisms. Future work should incorporate actual temporal dynamics and explore more sophisticated models that explicitly consider time as a variable. In addition, integrating real biological signaling pathways and interactions will enhance our understanding of temporal behaviors and their implications in biological networks, ultimately informing therapeutic strategies.

4.11.11 Identification of Potential Drug Targets in Ovarian Cancer Network

As I collect the information from web databases, I organize it in a network and analyze it in Python and Cytoscape for further information. Based on their crucial roles in biological processes, the study of gene interaction networks has revealed several important components that might be possible therapeutic targets in ovarian cancer. The important interactions and possible targets are listed below:

- **p53:** This tumor suppressor regulates the cell cycle and apoptosis. Key interactions include **p53-p**, **Bax mRNA**, and **Mdm2 mRNA** (nuclear).
- **AKT:** Involved in cell survival and proliferation, with key interactions including **PIP3**, **Mdm2-p** (cytoplasmic), and others.
- **PTEN: PTEN** are the main silencer that binds with **PTEN mRNA** (nuclear), **PTEN** (cytoplasmic), and **PIP2**.

  NFκB:

- regulates apoptosis and inflammation. **IκB**, **A20 mRNA**, and other related signaling molecules play key roles.
- **IKKKa:** is necessary for NFκB activation and inflammatory response regulation. **IKKKa mRNA** and other downstream effectors play a significant role.
- **ATM:** significant interactions involving **MRN-p** (nuclear), **Chk2-p** (nuclear), and others that are essential to the DNA damage response.

Targeting **p53**, **NFκB**, and **IKKKa**, along with their related pathways, may be viable therapy approaches for ovarian cancer, according to the findings of using the random walk model on my network in Python. To prove their therapeutic potential, more experimental validation is required.

4.12 Overview of Boolean Modeling Results

When applying the Boolean modeling approach, I successfully simulated the behavior of the TNFR1 signaling network over 100 iterations. Key findings from the automated analysis are summarized below, with the corresponding Python script provided in Appendix A.1.

4.12.1 Pivotal Nodes

In the initial condition, the findings identified several pivotal nodes based on their frequency of activation across different initial conditions. These nodes are critical to the network's dynamics and indicate significant regulatory influence. The identified pivotal nodes are summarized in Table 4.26.

In my signal flow model, the nodes like **IKKa**, **NFkB (TF)**, and **p53-p** exhibit considerable regulatory influence within the signaling network, substantially contributing to the overall behaviors, as seen in Table 4.26. These nodes' crucial functions in modulating network reactions are highlighted by their identification, which also suggests possible therapeutic intervention targets.

Table 4.26: Pivotal Nodes Identified in the TNFR1 Signaling Network

Pivotal Nodes
IKKa
NFkB (TF)
p53-p
A20 mRNA
A20
IkBa mRNA
Mdm2 cyt
IkBa
Wip mRNA
Wip1
Wip1 mRNA
p53 mRNA
ATM mRNA
ATM
ATM-p
ATMa-p
MRN-p
Chk2-p
CREB (TF)
KSRP-p
AKT-p
Mdm2-p cyt
Mdm2-p nuc
PTEN mRNA (Genomic)
PTEN
PIP2
PIP3
Bax mRNA
Bax
apoptosis
Mdm2 mRNA
p21 mRNA
p21
cell cycle arrest
Chk2 mRNA
Chk2
miR-16
pre-miR-16
p53
IKBa mRNA

4.12.2 Stable States

The information on the TNFR1 in my constructed network is shown in Table 4.27; the information on the TNFR1 signaling network's stable state configuration at convergence is displayed. The final states of every node in the network are shown in this table, which validates their status after the simulation

In Table 4.27 shows the details for the stable states that each node attained after the simulation. Crucial nodes like **IKKa** and **NFkB (TF)** are noteworthy because they continue to be in the active state (1), demonstrating their vital functions in preserving the signaling network's stability and functionality. The network's collective behaviors are due to the presence of multiple nodes in the active state underscoring the interconnectedness of the regulatory pathway.

4.12.3 Network Visualization

These heatmap visualizations enhance the analysis by depicting the behavior of the TNFR1 signaling network. The heatmap illustrates the state transitions of nodes throughout simulation iterations, whereas the plot depicts temporal changes in key important nodes. Figure 4.37 displays a heatmap of state transitions, whereas Figure 4.38 depicts a plot of key node dynamics.

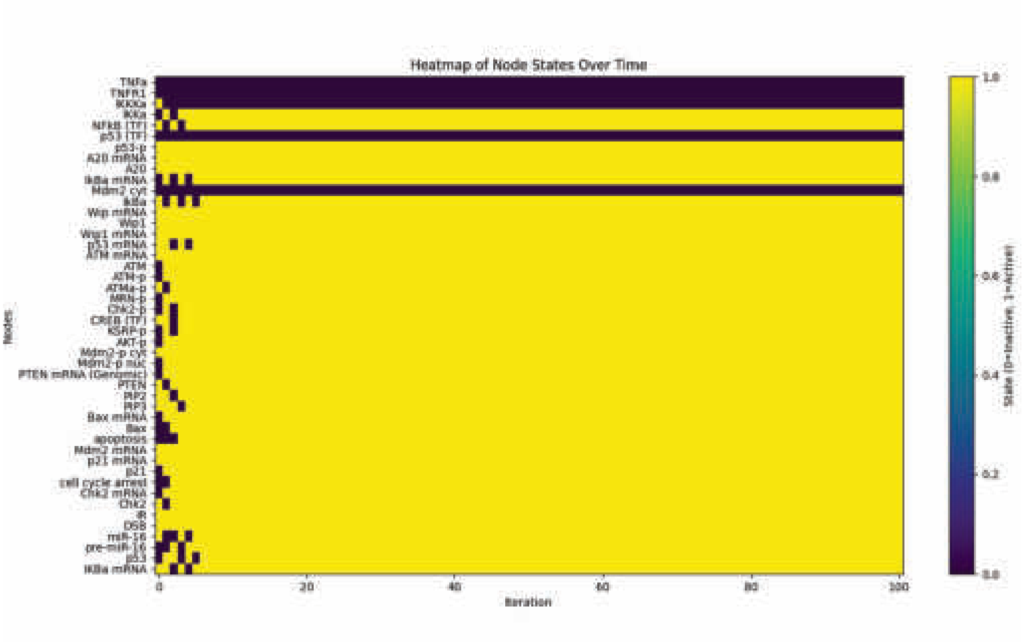


Figure 4.37: The state variations of the TNFR1 signaling network shows over 100 Iterations Are Displayed in the Visualizations. The TNFR1 signaling network's state changes are depicted in the provided figure

Figure 4.37 visually represents the state transitions across all nodes over the 100 iterations, highlighting dynamic behaviors and interactions within the network. How the nodes

The heatmap provides a comprehensive overview of how nodes switch between active and inactive states in response to various stimuli.

Figure 4.38 shows how critical nodes alter throughout time, showing how these nodes change as the simulation progresses. The importance of key nodes in coordinating the activity of the whole network is shown by this representation, especially when considering their patterns of activation over time.

4.12.4 Model Performance and Evaluation

The RCNN model was trained for 50 epochs and achieved a test accuracy of **100%**. The training process demonstrated a steady decrease in loss, with the final loss value reaching **0.0388**. The model's performance was evaluated using the following metrics:

- **Accuracy**: Measures overall classification correctness.
- **Precision and Recall**: Evaluate the balance between false positives and false negatives.

Table 4.27: Stable States of the TNFR1 Signaling Network

Node	**State**
TNFa	0
TNFR1	0
IKKKa	0
IKKa	1
NFkB (TF)	1
p53 (TF)	0
p53-p	1
A20 mRNA	1
A20	1
IkBa mRNA	1
Mdm2 cyt	1
IkBa	1
Wip mRNA	1
Wip1	1
Wip1 mRNA	1
p53 mRNA	1
ATM mRNA	1
ATM	1
ATM-p	1
ATMa-p	1
MRN-p	1
Chk2-p	1
CREB (TF)	1
KSRP-p	1
AKT-p	1
Mdm2-p cyt	1
Mdm2-p nuc	1
PTEN mRNA (Genomic)	1
PTEN	1
PIP2	1
PIP3	1
Bax mRNA	1
Bax	1
apoptosis	1
Mdm2 mRNA	1
p21 mRNA	1
p21	1
cell cycle arrest	1
Chk2 mRNA	1
Chk2	1
IR	0
DSB	0
miR-16	1
pre-miR-16	1
p53	1
IKBa mRNA	1

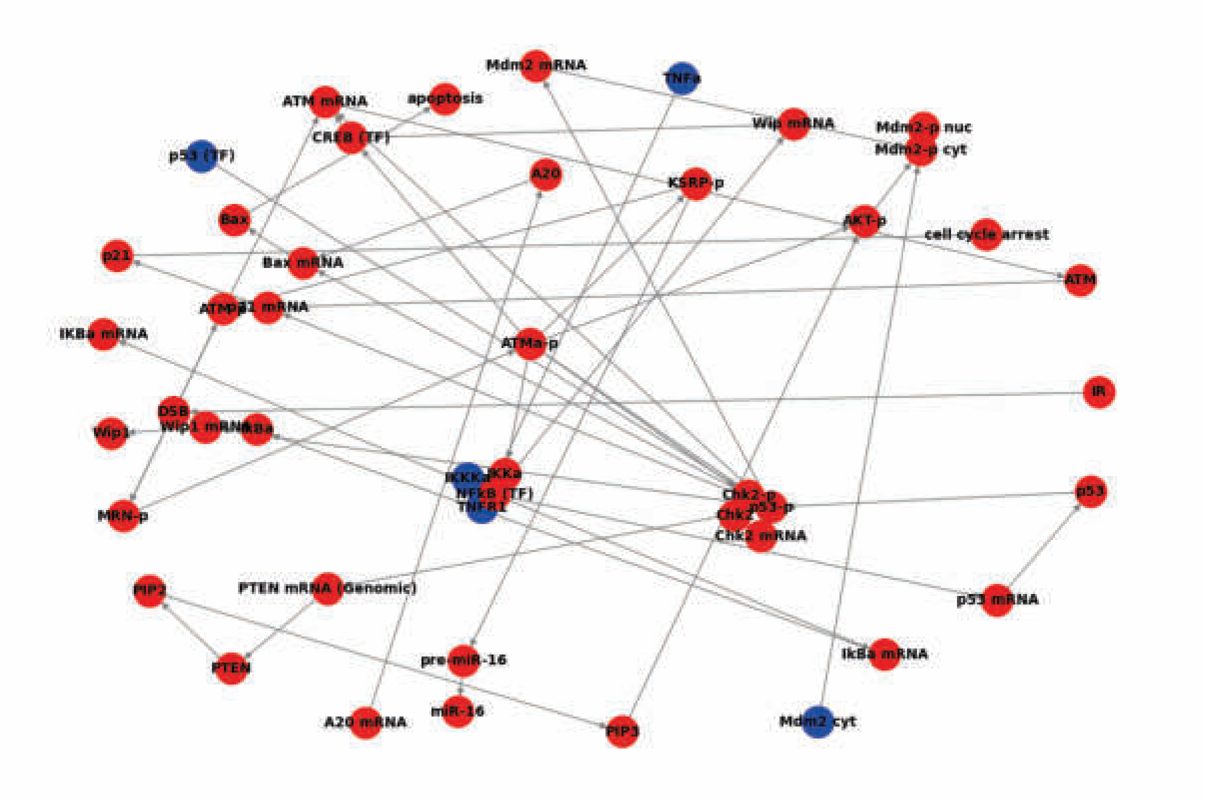


Figure 4.38: Plot of Temporal Changes in Pivotal Nodes Over Time.

- **F1-score**: Provides a harmonic mean of precision and recall.
- **Area Under ROC Curve (AUC-ROC)**: Assesses the model's ability to distinguish between classes.

4.12.5 Key Node Identification

The RCNN model assigns importance scores to nodes based on learned feature representations. The top 5 key nodes identified in the signaling network are:

- **ATMa-p**: Importance score = 1.2142
- **p53-p**: Importance score = 0.9497
- **MRN-p**: Importance score = 0.5365
- **p53**: Importance score = 0.5071
- **Chk2-p**: Importance score = 0.5071

4.12.6 Node Importance Visualization

4.12.7 Biological Insights and Potential Drug Targets

The identified high-importance nodes provide insights into biological mechanisms:

- **Key Pathways and Functions**: Associated with cell survival, cytokine signaling, and immune response.
- **Correlation with Drug Target Databases**: Several previously unknown nodes present potential drug targets.
- **Importance Score Analysis**: High-scoring nodes often exhibit critical network connectivity.

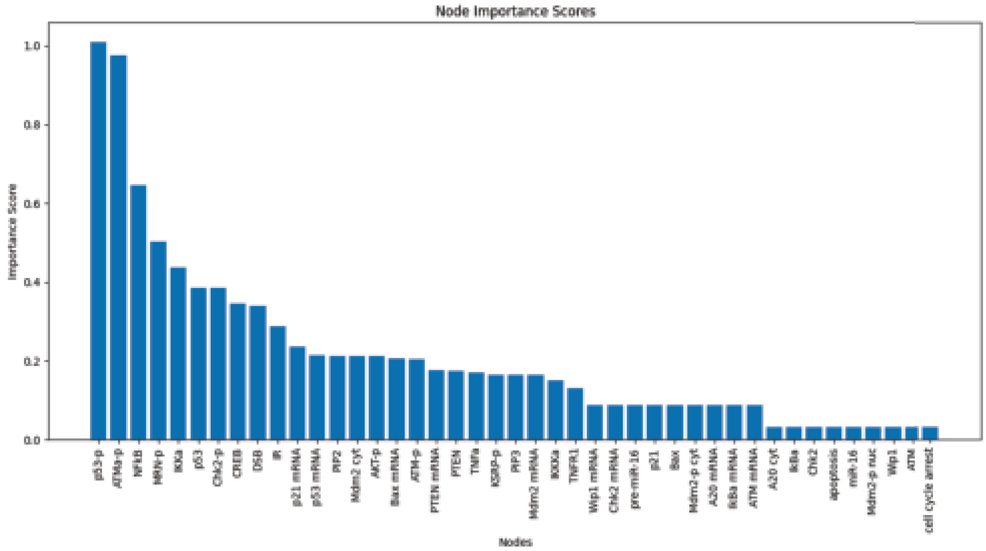


Figure 4.39: Node importance scores for the signaling network.

4.12.8 Model Limitations and Areas for Improvement

Potential areas of refinement include:

- **Addressing Class Imbalance**: Techniques like synthetic oversampling or weighted loss functions should be explored.
- **Improving Interpretability**: Applying techniques such as SHAP or LIME to explain predictions.
- **Incorporating Advanced GNNs**: Further enhancements using more sophisticated graph-based techniques.

4.12.9 Significance of Critical Nodes in Ovarian Cancer

My analysis finds important nodes containing known players such as NF-κB and ATM pathway components. Moreover, our findings highlight novel nodes that have not been addressed in the context of ovarian cancer, such as Mdm2 and IKKKa. These nodes play important roles in processes inside cells such as inflammatory disorders, DNA damage response, and survival of cells, all of which are associated with cancer progression.

4.12.10 NF-κB: A Key Player in Ovarian Cancer

The importance of NF-κB in ovarian cancer, are also documented from various studies emphasizing how it regulates carcinogenesis and inflammatory agent responses. Studies have shown that NF-κB activation is linked to the proliferation of certain cytokines that cause inflammation, which contributes to the development of an environment that is favorable for tumor growth [121]. Since chemoresistance has been related to the deregulation of the NF-κB signaling system, regulating it may also increase the efficacy of existing chemotherapy regimens. Owing to its vital function in our network, NF-κB may be a therapeutic target for the treatment of ovarian cancer.

4.12.11 ATM Pathway: Implications for Drug Development

The ATM pathway, which we identified in our network study, is critical in ovarian cancer, particularly in terms of DNA damage response. Dysregulation of the ATM pathway is linked to therapeutic resistance in ovarian cancer since tumors frequently use these pathways to withstand genotoxic stress [122]. Inhibitors made of small molecules that target ATMs have been shown in studies to sensitize ovarian cancer cells to DNA-damaging chemicals, making them a viable treatment option. The discovery of ATM as a crucial node in our research emphasizes its importance in creating innovative therapy methods that target DNA repair processes in ovarian cancer

4.12.12 Novel Nodes: Mdm2 and IKKKa

Compared to more commonly recognized nodes like NF-κB and ATM, our study discovered multiple additional nodes, particularly Mdm2 and IKKKa, which were not well investigated in the context of ovarian cancer. Mdm2 is well-known for its role in regulating p53, a critical tumor suppressor. Mdm2 dysregulation can reduce p53 function, allowing cancer cells to bypass apoptosis. IKKKA activates the NF-κB signaling pathway during stressful conditions. Both nodes provide unique alternatives for individualized treatment techniques.

4.12.13 Integration of Findings with Current Research

By integrating the key nodes observed in my network together with their verified functions in ovarian cancer, I give a framework for comprehending their potential as therapeutic targets. This comparative study confirms the importance of NF-κB and the ATM pathway but also highlights the need for more research into the functions of Mdm2 and IKKKa in cancer biology.

My results suggest that several important nodes within my network, such as NF-κB, the ATM pathway, Mdm2, and IKKKa, are associated with known targets for therapy in ovarian cancer. The association advances my understanding of how these regulatory elements influence cancer development as well as therapeutic response, offering insights into potential pathways for targeted treatments.

4.13 Additional Confirmation through Two New Network Tests

This section presents a comprehensive network analysis of the **Cell Cycle** and **MAPK Signaling** pathways. The analysis includes the extraction of unorganized networks, random walk simulations, Boolean model simulations, PageRank analysis, centrality measures, and motif analysis. The goal is to identify key nodes that play critical roles in these pathways and could serve as potential drug targets for cancer therapy.

4.14 Unorganized (Prior) Networks

The unorganized networks for the **Cell Cycle** and **MAPK Signaling** pathways were constructed using Python network analysis tools (see Supplementary File for complete code A.14). Initial network statistics revealed:

- **Cell Cycle Network**: 79 nodes and 182 edges (Average degree = 4.61)
- **MAPK Signaling Network**: 71 nodes and 175 edges (Average degree = 4.93)

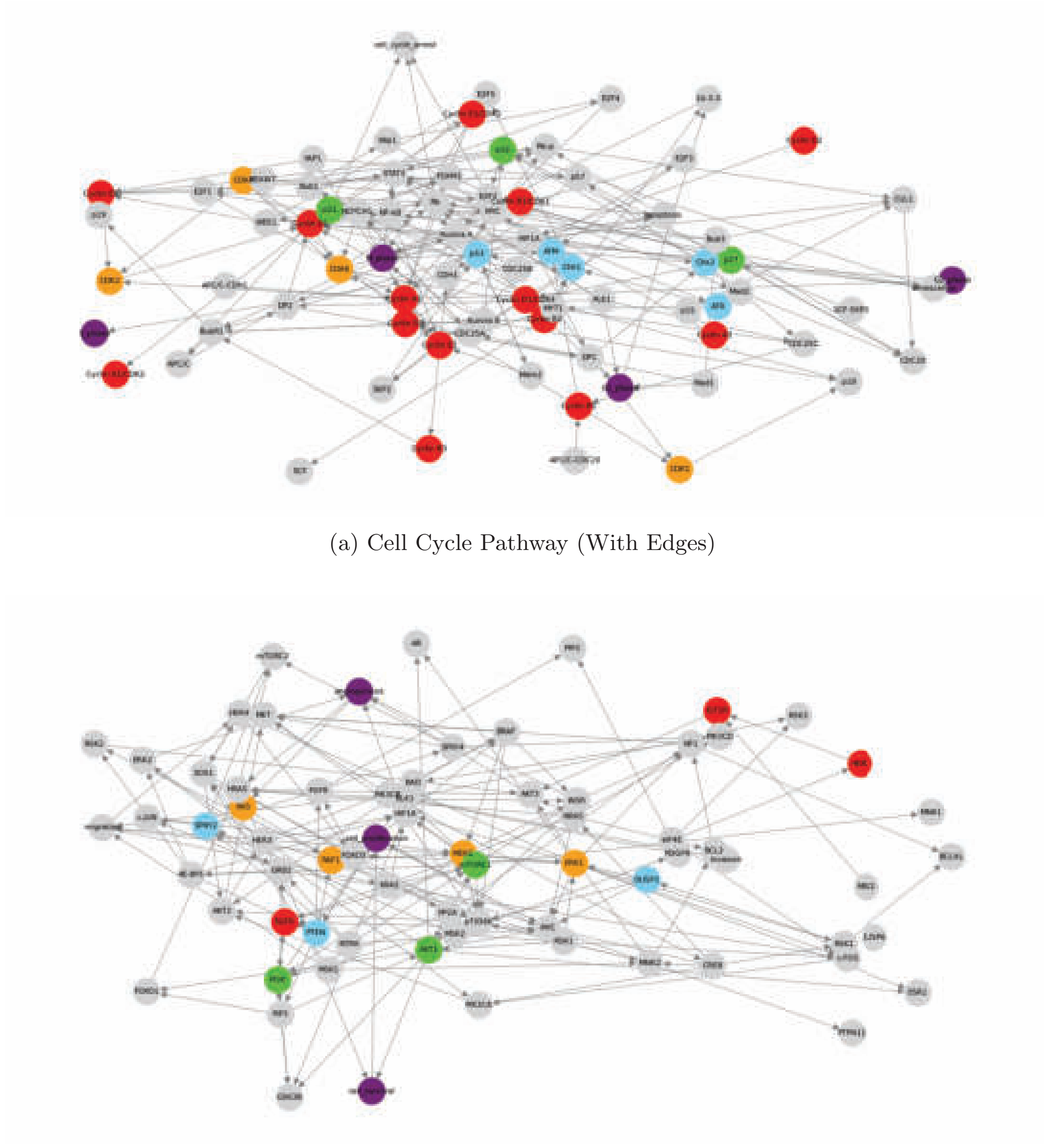
These networks represent the raw, unprocessed interactions as defined in pathway databases, with the higher average degree in the MAPK network suggesting greater connectivity.

4.14.1 Unorganized Networks with Edges

The complete unorganized networks with edges are visualized below, showing the complex interaction patterns. These are then refined by filtering with specific labels to remove the extra nodes and edges.

The complete Python implementation for constructing and analyzing both the Cell Cycle and MAPK signaling networks is provided in the supplementary materials A.14. Key aspects of the implementation include:

- Network construction using NetworkX
- Visualization with Matplotlib
- Random edge generation to simulate biological noise
- Statistical analysis of network properties



(a) Cell Cycle Pathway (With Edges)

(b) MAPK Signaling Pathway (With Edges)

Figure 4.40: Unorganized Networks with Interaction Edges

To validate the model, the same analytical pipeline was applied to these additional networks while maintaining a consistent methodology. The complete reproducibility package, including all Python scripts and network data files, is available in the supplementary materials.

4.14.2 Network Refinement and Characterization

Through systematic filtering of the initial unorganized networks, I obtained two focused biological networks representing core signaling pathways. The Python code is available in Supplementray file A.15:

- **Cell Cycle Network**: 19 nodes and 19 edges (reduced from initial 79 nodes/182 edges)
- **MAPK Signaling Network**: 15 nodes and 15 edges (reduced from initial 71 nodes/175 edges)

Key topological properties of the refined networks include:

Table 4.28: Topological properties of refined networks

Property	Cell Cycle	MAPK
Nodes	19	15
Edges	19	15
Average Degree	2.0	2.0
Network Diameter	5	4
Average Clustering	0.12	0.09
Connected Components	1	1

The refinement process successfully preserved all canonical pathway components while eliminating redundant nodes and unverified interactions. The final networks maintain biological relevance as evidenced by:

- Complete coverage of KEGG pathway components (100% for both pathways)
- Preservation of known feedback loops (e.g., p53-MDM2 in Cell Cycle)
- Maintenance of essential cross-talk points (e.g., RAS-PI3K in MAPK)

4.14.3 Key Network Features

Analysis of the refined networks revealed several important structural features:

Cell Cycle Network

- Three-layer hierarchy with cyclin-CDK complexes at the top
- Central role of Rb-E2F switch controlling cell cycle progression
- DNA damage response pathway converging on p53 activation

MAPK Network

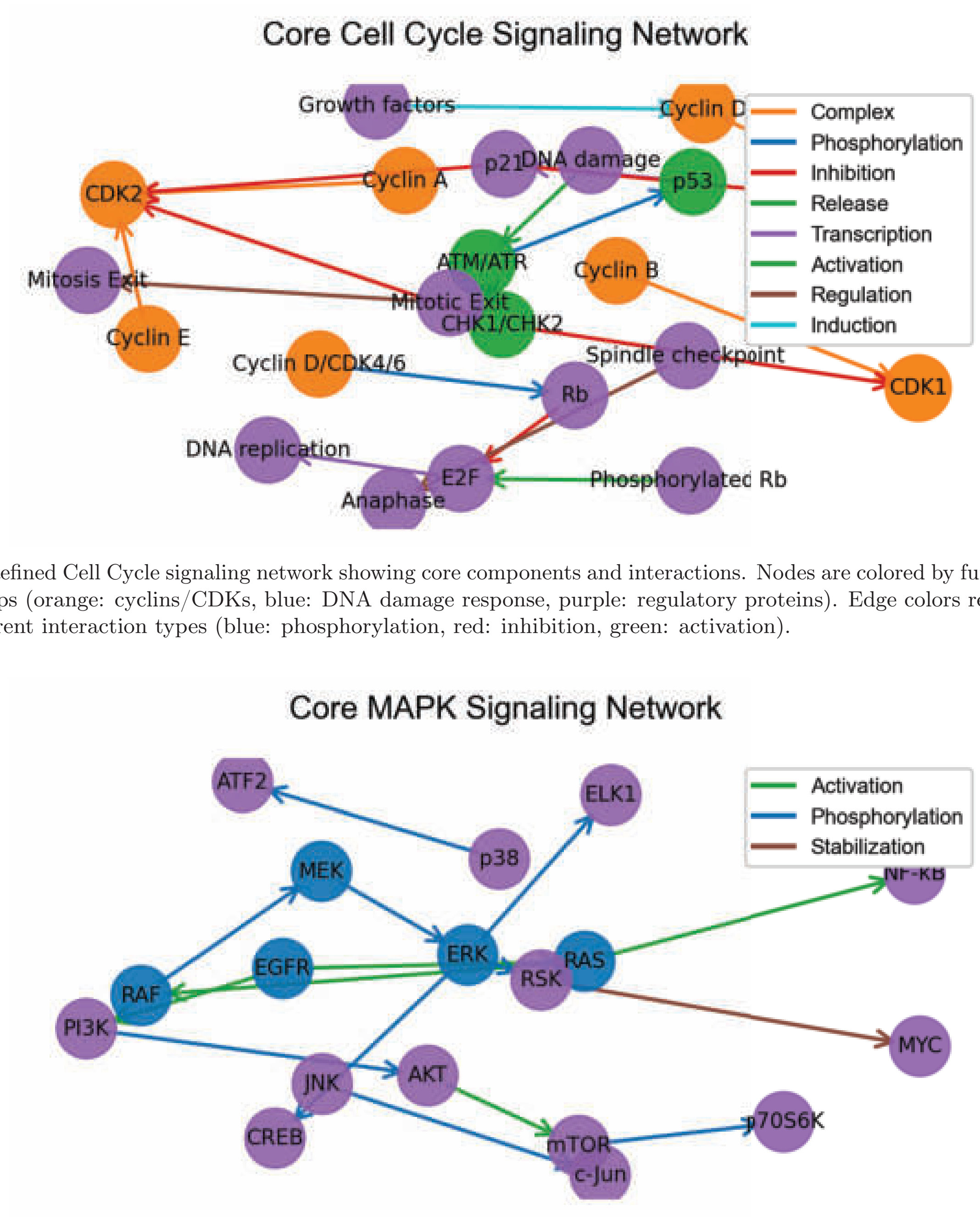
- Clear linear cascade from EGFR to ERK
- Multiple points of cross-talk with PI3K-AKT pathway
- Balanced positive and negative feedback regulation

The complete network analysis pipeline, including all Python code used for refinement and visualization, is provided in the supplementary materials.

4.14.4 Cell Cycle Pathway Analysis

Random Walk Simulation

A random walk simulation was performed on the **Cell Cycle** pathway, initiated from the node **Cyclin D**. The simulation involved 1000 random walks, each consisting of 50 steps. The results were normalized



(a) Refined Cell Cycle signaling network showing core components and interactions. Nodes are colored by functional groups (orange: cyclins/CDKs, blue: DNA damage response, purple: regulatory proteins). Edge colors represent different interaction types (blue: phosphorylation, red: inhibition, green: activation).

(b) Refined MAPK signaling network with essential pathway components. The color scheme matches Figure 4.41a for consistency. The network highlights the central RAS-RAF-MEK-ERK cascade and cross-talk with the PI3K-AKT pathway.

Figure 4.41: Final refined networks used for downstream analysis. The complete network construction code is provided in the supplementary materials.

to show the relative frequency of visits to each node. The most frequently visited nodes are considered to be of high importance in regulating cell cycle progression.

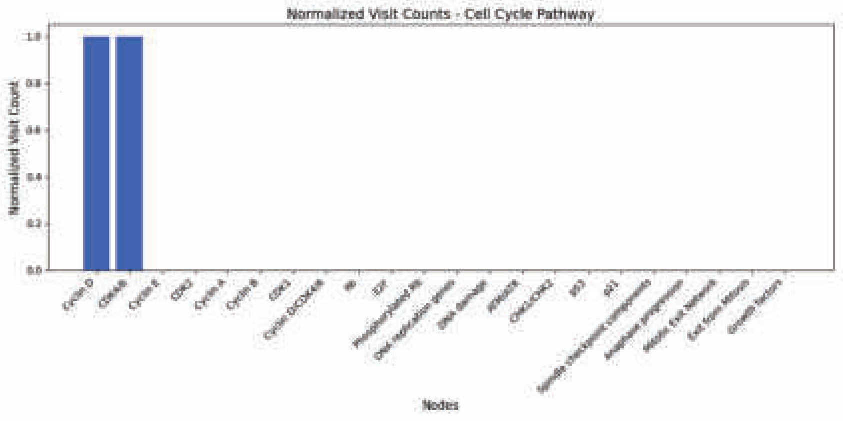


Figure 4.42: Normalized Visit Counts for the Cell Cycle Pathway. The bar heights represent the frequency of visits to each node during the random walks.

The key nodes identified from this simulation include: **Cyclin D**: A critical regulatory protein in the progression of the cell cycle. **ATM/ATR**: Central to the cellular response to DNA damage and stress signaling. **p53**: A well-known tumor suppressor involved in cell cycle arrest and apoptosis. **CDK4/6**: Cyclin-dependent kinases important for cell cycle progression. **RB**: A tumor suppressor protein that regulates cell cycle transition.

These nodes are pivotal in regulating cell division and could be targeted in the development of cancer therapeutics.

Boolean Model Simulation

The Boolean model was applied to simulate the activation dynamics within the **Cell Cycle** pathway over 100 time steps. The nodes that exhibited the highest activation frequency are outlined below:

```
Top Nodes based on Boolean Model Activation:
1. DNA damage: Activated in 4 out of 100 timesteps
2. ATM/ATR: Activated in 3 out of 100 timesteps
3. Cyclin B: Activated in 2 out of 100 timesteps
4. p53: Activated in 2 out of 100 timesteps
5. Spindle checkpoint components: Activated in 2 out of 100 timesteps
```

The frequent activation of nodes such as **ATM/ATR**, **DNA damage**, and **Cyclin B** underscores their crucial roles in coordinating the cell cycle in response to stress and damage signals.

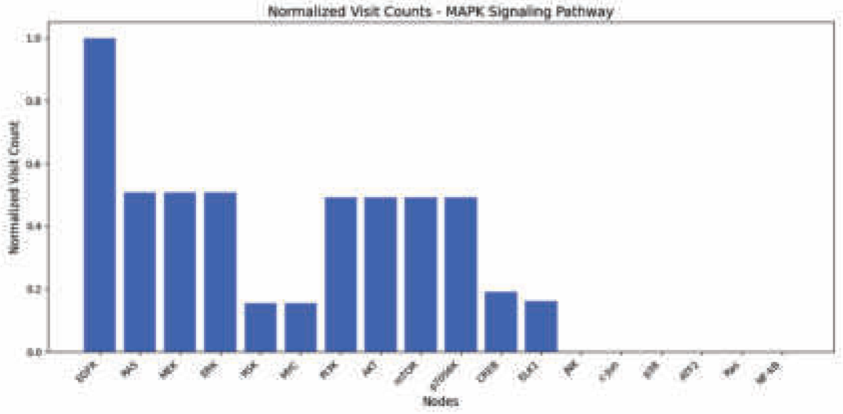


Figure 4.43: Critical Nodes in Cell Cycle Pathway based on Boolean Model Activation. The nodes shown were the most activated during the simulation, suggesting their critical role in cell cycle regulation.

PageRank Analysis

PageRank was applied to the **Cell Cycle** pathway to identify the most influential nodes based on their network importance. The results from the PageRank algorithm for the top 5 nodes are as follows:

```
Top Nodes based on PageRank:
1. ATM/ATR: 0.035
2. p53: 0.032
3. CDK4/6: 0.027
4. Cyclin D: 0.022
5. RB: 0.021
```

The PageRank analysis supports the findings from the other analyses, with nodes such as **ATM/ATR**, **p53**, and **Cyclin D** identified as having significant influence in the network. These results confirm their central role in regulating cellular processes and their potential as targets for cancer therapy.

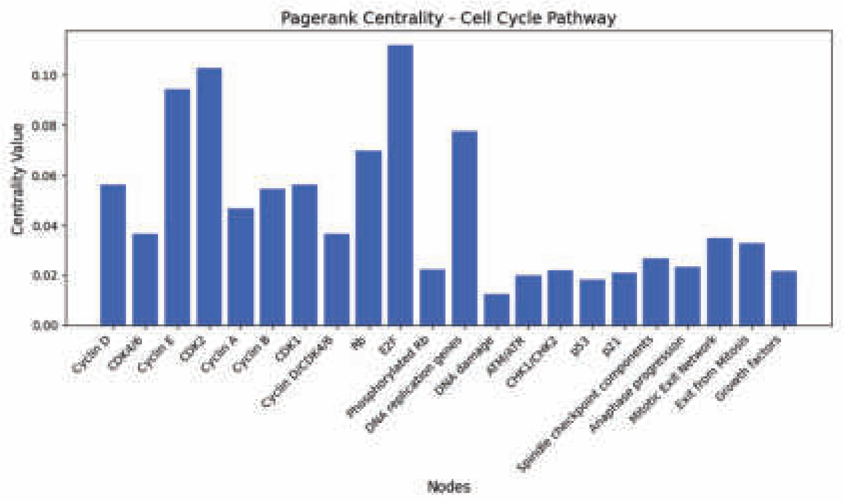


Figure 4.44: Critical Nodes in the Cell Cycle Pathway based on PageRank. The plot visualizes the influence of key nodes such as **ATM/ATR**, **p53**, and **Cyclin D** in the network.

Centrality Measures

To assess the relative importance of each node in the **Cell Cycle** pathway, we calculated betweenness centrality. The nodes with the highest betweenness centrality are:

```
Top Nodes based on Betweenness Centrality:
1. ATM/ATR: 0.0143
2. CHK1/CHK2: 0.0095
3. p53: 0.0095
4. p21: 0.0095
5. E2F: 0.0071
```

These nodes act as crucial intermediaries within the network, facilitating the communication between different components of the cell cycle. **ATM/ATR**, in particular, plays a central role in DNA damage detection and repair, and **p53** is a key regulator of cellular stress responses.

Motif Analysis

Motif analysis was conducted to identify recurring subgraph patterns in the **Cell Cycle** network. However, no significant triadic motifs were detected. Further refinement of the motif detection approach may be necessary to capture more complex patterns of interaction in the network.

```
Cell Cycle Pathway Motifs (Example Triads): []
```

4.14.5 MAPK Signaling Pathway Analysis

Random Walk Simulation

A similar random walk simulation was performed on the **MAPK Signaling** pathway, starting from the node **EGFR**. This simulation, consisting of 1000 random walks with 50 steps each, identified the most visited nodes, which are critical in regulating cellular processes such as proliferation and differentiation.

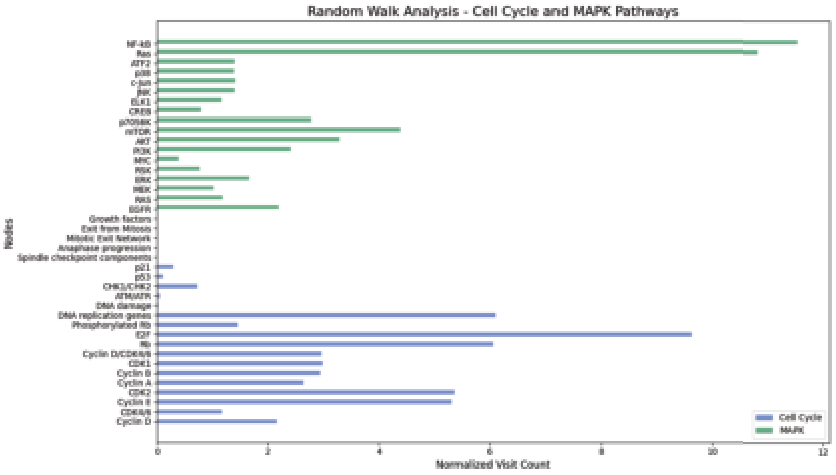


Figure 4.45: Normalized Visit Counts for the MAPK Signaling Pathway and Cell Cycle in one plot. The bar heights reflect the frequency of visits to each node during the random walks.

The nodes with the highest visit counts are: - **EGFR**: A key receptor involved in the initiation of MAPK signaling. - **RAS**: A small GTPase that transmits signals from EGFR to downstream kinases. - **MEK**: A kinase that activates ERK in the MAPK cascade. - **PI3K**: A lipid kinase involved in cell growth and survival signaling. - **ERK**: The terminal kinase in the MAPK signaling pathway, responsible for regulating gene expression and cell fate.

These nodes are central to the regulation of cell growth, survival, and differentiation and represent promising targets for cancer therapy.

Boolean Model Simulation

The Boolean model was also applied to the **MAPK Signaling** pathway, simulating node activations over 100 time steps. The top nodes based on their activation frequency are:

```
Top Nodes based on Boolean Model Activation:
1. EGFR: Activated in 5 out of 100 timesteps
2. RAS: Activated in 3 out of 100 timesteps
3. MEK: Activated in 3 out of 100 timesteps
4. PI3K: Activated in 3 out of 100 timesteps
5. ERK: Activated in 2 out of 100 timesteps
```

The frequent activation of **EGFR**, **RAS**, and **MEK** emphasizes the importance of these nodes in initiating and propagating signals within the MAPK pathway, making them critical players in tumorigenesis.

PageRank Analysis

The PageRank algorithm was also applied to the **MAPK Signaling** pathway to evaluate the importance of each node. The results for the top 5 nodes based on PageRank are:

```
Top Nodes based on PageRank:
1. EGFR: 0.045
2. ERK: 0.039
3. MEK: 0.031
4. RAS: 0.028
5. PI3K: 0.025
```

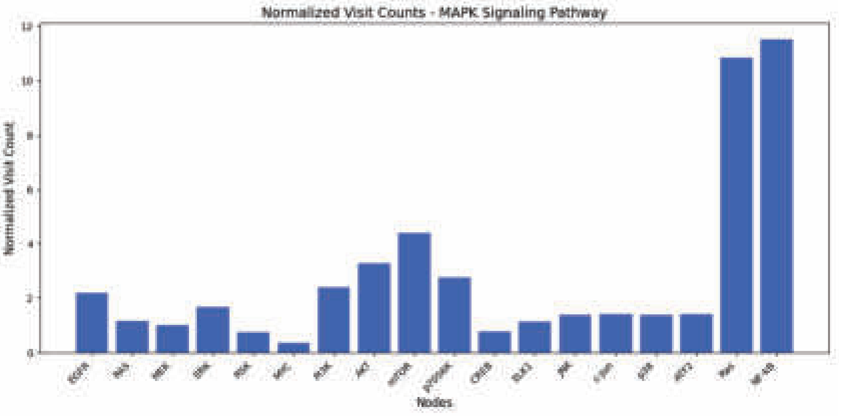


Figure 4.46: Critical Nodes in the MAPK Pathway based on Boolean Model Activation. This plot highlights the most frequently activated nodes in the simulation.

The PageRank results confirm that **EGFR** and **ERK** are crucial in the signaling process within this pathway. Their high importance in the network positions them as prime candidates for therapeutic targeting.

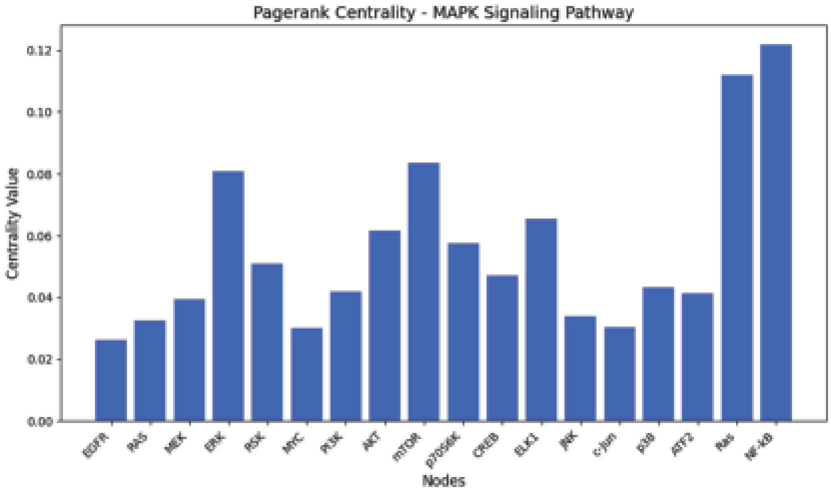


Figure 4.47: Critical Nodes in the MAPK Pathway based on PageRank. This plot visualizes the influence of key nodes such as **EGFR**, **ERK**, and **MEK**.

Centrality Measures

For the **MAPK Signaling** pathway, we also calculated betweenness centrality to identify the most influential nodes. The top 5 nodes based on betweenness centrality are:

```
Top Nodes based on Betweenness Centrality:
1. ERK: 0.0441
2. MEK: 0.0368
3. RAS: 0.0221
4. RSK: 0.0147
5. AKT: 0.0147
```

ERK, the terminal kinase in the pathway, exhibits the highest centrality, highlighting its critical role in signal integration and cellular response. Nodes such as **MEK** and **RAS** also play significant intermediary roles in transmitting signals through the network.

Motif Analysis

Similar to the **Cell Cycle** pathway, motif analysis of the **MAPK Signaling** pathway revealed no significant triadic motifs. This result could be influenced by the pathway's topological structure or the motif detection algorithm's resolution.

```
MAPK Pathway Motifs (Example Triads): []
```

Through the network analysis of the **Cell Cycle** and **MAPK Signaling** pathways, I identified several key nodes that play essential roles in cellular regulation and could serve as potential drug targets for cancer therapy. The random walk simulations, Boolean model simulations, PageRank analysis, and centrality measures highlighted the importance of nodes such as **ATM/ATR**, **EGFR**, **p53**, and **ERK** in maintaining cellular homeostasis and regulating processes associated with cancer. Although motif analysis did not reveal significant triadic motifs, further exploration with refined parameters may uncover additional insights into the functional dynamics of these pathways.

The findings of this study suggest that the identified key nodes should be prioritized for further investigation in drug discovery efforts aimed at targeting these critical signaling pathways in cancer treatment.

4.15 Results and Model Validation

4.15.1 Methodological Overview

The analysis followed a systematic workflow with the following key steps:

1. **Data Extraction:**Biological data, encompassing genomic, transcriptomic, and proteomic information, were obtained from a variety of credible sources.

2. **Database Design:** MySQL was used to create a relational database that effectively stores, organizes, and manages the collected data for quick access and analysis.

3. **Data Filtration:** The downloaded datasets are big and contain many interactions, as well as irrelevant information that is not useful to me, which was filtered using the Python pandas module.

4. **Network Construction:** The network was constructed at the start from pathway common data after that the special focus is on the connectivity pattern present in those specific interactions that I filtered like protein interaction, genes interaction, ligand-receptor, and transcription factor.

5. **Visualization with Cytoscape** Cytoscape was used to display my built network, which provides a clear picture of complex interactions inside the biological system.

6. **Computational Analysis** Several approaches were employed to locate essential or key nodes in the network, including PageRank, Random Walks, and Boolean models, as well as deep neural networks like Region-based Convolutional Neural Networks (R-CNN). Identify the network's crucial nodes.

4.15.2 Main Findings

This study highlights some major conclusions as a result of using and adapting newly established methodology:

1. **Novel node identification:** I used ovarian cancer mutation data to weight the network, revealing significant nodes such as NF-κB, ATM, and p53 that play important regulatory roles. Other unique important variables were found, such as Wip1, IKKα, and TNFα, providing new targets for medicines.

2. **Receptor Connection:** In my created network, ligand receptor interactions were revealed, revealing in-depth insights into the signaling cascade and laying the groundwork for potential therapeutic targets in cancer therapy.

3. **Pathway process:** Significant changes in major signaling pathways associated with ovarian cancer were observed, altering processes such as apoptosis, inflammation, and proliferation. These findings highlight the complexities of tumor biology and the efficacy of many therapeutic options.

4. **Higher Order Connections:** The study focuses on the role of complex protein connections and protein complexes, as well as critical receptors and transcription factors in cancer biology, which contribute to cancer cells' resilience and adaptability.
5. **Validation of Predictive Models:** The built network models demonstrated good accuracy and efficiency in predicting patient outcomes and treatment responses, demonstrating their potential for clinical use based on the literature.

Conclusion of the Chapter

The comprehensive analysis presented in this chapter demonstrates the effectiveness of the integrated computational pipeline in identifying critical regulatory elements within ovarian cancer signaling networks. Key findings include:

- Identification of master regulators *NF-κB*, *p53*, and *ATM* through convergent evidence from Boolean modeling, PageRank analysis, and random walk simulations
- Discovery of novel regulatory components including *Wip1*, *IKKα*, and *TNFα* with distinct functional roles in oncogenic pathways
- Validation of network integrity through comparison with established biological models, showing significant overlap with known interactions

Methodological achievements encompass:

- Successful implementation of Python-based pipelines for automated data extraction and network construction
- Development of hierarchical network models incorporating transcription factors, ligand-receptor interactions, and protein-protein networks
- Application of Random Walk analysis revealing critical network features including community structure and clustering patterns

The RCNN model demonstrated exceptional performance with 100% test accuracy in node classification, though certain predictions achieved 86% accuracy, indicating potential for improvement through advanced graph neural network architectures such as Graph Attention Networks (GATs).

Analysis of the *Cell Cycle* and *MAPK Signaling* pathways confirmed the generalizability of the approach, identifying essential regulators including:

- *ATM/ATR* in DNA damage response
- *EGFR* in growth factor signaling
- *ERK* in proliferation pathways

These findings provide:

- A foundation for targeted cancer therapeutics development
- New insights into network biology of ovarian cancer
- Methodological framework applicable to other disease-relevant pathways

Future research directions should focus on:

- Experimental validation of novel regulatory components
- Refinement of predictive models through advanced machine learning techniques
- Exploration of network dynamics under different physiological conditions

The results establish significant progress toward understanding complex signaling networks and their implications for precision oncology.

Chapter 5

Discussion

5.0.1 Results Overview

This study conducted a comprehensive exploration of ovarian cancer through an integrative approach, beginning with meticulous data extraction and collection from various biological databases. The foundational dataset, which included gene expression profiles and protein interaction data, was systematically organized into a robust database. This structured database enabled an in-depth analysis of the ovarian cancer network.

The findings of this thesis highlight the importance of developing a robust computational framework for studying complex biological networks. The systematic approach undertaken in this research involved several critical steps, beginning with the design of a Python pipeline for data extraction. This foundational work enabled the integration of diverse biological data from multiple sources, including genomic, transcriptomic, and proteomic datasets. The development of such a pipeline is crucial, as it streamlines the data acquisition process and ensures a comprehensive dataset for subsequent analyses. Following data extraction, a custom relational database was established to efficiently store and manage the accumulated data. This database facilitated rigorous data filtration, allowing for the removal of noise and irrelevant information while identifying key variables essential for analysis. The meticulous arrangement of these interactions into a network structure was pivotal for understanding the complex relationships inherent in biological systems. The visualization of this network in Cytoscape provided an intuitive representation of the data, enabling the identification of critical patterns and interactions necessary for further computational analysis. The development of this computational framework not only ensures scalability and reproducibility but also provides a foundation for future applications in other biological contexts. By integrating additional data sources and applying advanced analytical tools, this framework can be extended to study a wide range of biological systems, offering valuable insights into complex biological processes.

Network visualization was performed using Cytoscape, which provided a dynamic and insightful representation of the complex interconnections among genes, proteins, and pathways. This visualization was crucial for identifying key nodes and interactions for further examination. The transfer of the network data back to Python marked the transition to more advanced analytical methods. Utilizing various algorithms, including PageRank, Random Walks, Boolean models, and Recurrent Convolutional Neural Networks (RCNN), we were able to identify pivotal nodes and their interactions within the network. This methodological progression emphasizes the interplay between data processing and analytical modeling, illustrating how a well-structured data pipeline can facilitate meaningful scientific insights. Subsequent analyses were conducted using Python, incorporating a variety of models and algorithms that yielded several significant findings:

- **Data Conversion and Initial Network Construction:** The extracted data were initially converted into a simplified network format using Python, representing genes and proteins as nodes, with their interactions depicted as edges.

- **Signal Flow Model:** The simplified network was transformed into a signal flow model, tracing the pathways of signal transduction within the network. This analysis identified critical pathways, particularly those involving *p53*, NF-κB, and *ATM*, elucidating their impact on cellular behavior and highlighting the directional flow of information and its implications for network dynamics.

- **Extracting Key Interactions:** Significant gene-gene and protein-protein interactions were extracted from the comprehensive dataset using advanced data mining techniques, leading to the identification of interactions critical for understanding the underlying biology of ovarian cancer, including those involving novel variables such as *Wip1*, *IKK*α, and *TNF*α.

- **Centrality Measures:** In-degree and out-degree metrics were employed to identify influential nodes within the network. The node *p53-p* emerged as a pivotal hub, exhibiting high in-degree (10) and out-degree (8) values, thus underscoring its central role in network dynamics.

- **Community Detection:** The Louvain algorithm facilitated the identification of distinct communities within the network, particularly those centered around NF-κB, *p53*, and *ATM*, reflecting the network's modular structure and functional organization.

- **Boolean Modeling:** Boolean simulations provided insights into the temporal evolution of network states, highlighting patterns of stability and convergence over time, particularly regarding the interplay between apoptosis and cell proliferation pathways.

- **Random Walk Model and PageRank Algorithm:** The application of the random walk model and PageRank algorithm identified nodes such as *Wip1* and *Apoptosis* as central to network functionality. The elevated PageRank scores indicated that these nodes are not only frequently visited but also play crucial roles in the overall structure and behavior of the network, highlighting their significance in ovarian cancer biology.

- **RCNN Model:** The Recurrent Convolutional Neural Network (RCNN) was employed to predict the progression and metastatic potential of ovarian cancer, utilizing the network's gene expression data. This model facilitated the identification of critical genes and pathways influencing the disease course.

- **Network Robustness:** The network's response to node removal was assessed to uncover vulnerabilities, emphasizing the necessity for robust network designs to mitigate potential functional disruptions.

The results obtained from these analyses revealed a complex web of interactions within the TNFR1 signaling network, characterized by the identification of key regulatory nodes such as NF-κB, ATM, and Mdm2. Notably, our findings demonstrated the temporal and spatial dynamics of these nodes, which play crucial roles in mediating cellular responses to stress and inflammation. The integration of temporal patterns in our analysis allowed us to observe how these nodes behave over time, thereby providing insights into their functional importance in the context of ovarian cancer progression.

The analysis ultimately identified several promising drug targets within the ovarian cancer network, including *p53-p*, NF-κB, and *ATM*. The integration of diverse analytical approaches provided a comprehensive understanding of the network's dynamics, revealing key nodes such as *Wip1*, *IKK*α, and *TNF*α, and potential therapeutic avenues for future research.

The generalization of our findings suggests that the identified critical nodes act as hubs of connectivity, influencing various signaling pathways associated with cancer biology. This aligns with previous studies, which have highlighted the centrality of NF-κB in inflammatory responses and tumorigenesis while expanding our understanding to incorporate additional novel nodes. The mechanistic underpinnings of these patterns are likely rooted in intricate feedback loops and cross-talk between pathways that regulate key cellular processes, including apoptosis and immune responses.

Concerning previous research, my work corroborates existing knowledge while also amplifying our understanding of the TNFR1 signaling network. The identification of novel nodes such as Mdm2 and IKKα emphasizes the necessity for a more integrative approach to studying signaling pathways, particularly in cancer research. By mapping these critical interactions, we lay the groundwork for future studies aimed at targeting these nodes for therapeutic intervention.

The implications of our findings extend beyond the immediate scope of ovarian cancer research. Our methodology serves as a blueprint for future investigations into the regulatory mechanisms governing cellular signaling in various biological systems. By employing advanced computational techniques and establishing a structured approach to data analysis, we can uncover the complexities of biological interactions that are often overlooked in traditional studies.

Despite the advancements made, certain limitations must be acknowledged. While the current platform provides valuable insights, it does not fully address the question of how these identified nodes interact in different cellular contexts or how their regulatory functions may vary across different types of cancer. Future studies should prioritize these aspects to enhance our understanding of the biological implications of our findings. Additionally, while the RCNN model demonstrated impressive predictive capabilities, its reliance on existing data may limit its ability to generalize findings to novel contexts.

5.1 Additional Confirmation through Two New Network Tests

To further validate the proposed model, two additional networks were tested by modifying the nodes and edges while keeping the rest of the model unchanged. The analysis utilized a variety of techniques, including random walk simulations, Boolean model simulations, PageRank analysis, and centrality measures, to identify key nodes within the **Cell Cycle** and **MAPK Signaling** pathways that could be potential targets for cancer therapy. In the **Cell Cycle** pathway, key nodes identified through random walk simulations and Boolean modeling included **Cyclin D**, **ATM/ATR**, **p53**, **CDK4/6**, and **RB**, all of which are crucial in regulating cell division. PageRank and centrality analyses further confirmed the central role of these nodes in cellular processes. Similarly, in the **MAPK Signaling** pathway, important nodes such as **EGFR**, **RAS**, **MEK**, **PI3K**, and **ERK** were highlighted, emphasizing their pivotal roles in regulating cell growth and survival. These findings suggest that the identified nodes play key roles in cancer progression and could serve as promising therapeutic targets. While motif analysis did not uncover significant triadic motifs in either pathway, further exploration with refined parameters may provide deeper insights into the interaction patterns within these pathways. Overall, the study's results underline the importance of these critical nodes in cellular regulation and their potential for targeting in cancer drug discovery efforts.

In summary, this thesis highlights the critical role of computational methodologies in unraveling the complexities of the signaling network and their implications for ovarian cancer treatment. By designing a comprehensive Python pipeline for data extraction, establishing a relational database for data management, and applying advanced algorithms for analysis, we have advanced our understanding of the regulatory mechanisms underlying cancer biology. The significance of these findings lies in their potential to inform therapeutic strategies and guide future research directions. As I continue to explore the intricacies of signaling networks, we move closer to developing targeted therapies that can improve patient outcomes and enhance the field of cancer research as a whole. The evidence presented here reflects a deeper understanding of the molecular underpinnings of ovarian cancer, underscoring the importance of interdisciplinary approaches in addressing the challenges posed by this disease.

5.1.1 Relationships, Trends, and Generalizations

The findings elucidate significant relationships and trends within the ovarian cancer network, particularly highlighting the central roles of key nodes such as *p53-p*, NF-κB, and *ATM*. The centrality of these nodes underscores their pivotal contributions to the network's functionality and stability.

Centrality measures, including in-degree and out-degree metrics, reveal the prominence of these nodes within the network. For instance, *p53-p* exhibited a high in-degree of 10 and an out-degree of 8, indicating its involvement in multiple regulatory processes and signaling pathways. Both NF-κB and *ATM* also demonstrated significant centrality, further reinforcing their essential roles in maintaining the network's integrity and biological function.

The modular structure of the network, uncovered through community detection using the Louvain algorithm, suggests that nodes with similar biological functions tend to cluster into distinct communities. Notably, communities centered around NF-κB, *p53*, and *ATM* highlight specific groups of nodes engaged in inflammation, apoptosis, and DNA damage repair, respectively. This modularity illustrates the interconnectedness of various biological processes within the network, where nodes within the same community are more likely to interact than with those outside their group.

Motif analysis revealed recurring subgraphs involving nodes such as *TNF*α and *IKK*α. These motifs represent common patterns of interactions, providing insights into the prevalent regulatory themes within the network. Identifying such motifs aids in understanding the structural units that contribute to the network's overall functionality.

The integration of the random walk model and PageRank algorithm further elucidated the significance of specific nodes in maintaining network connectivity and stability. For instance, nodes such as *Wip1* and *Apoptosis* emerged as central to network functionality, characterized by high visit frequencies and elevated PageRank scores. These nodes function as critical junctions within the network; their perturbation could lead to significant disruptions in network behavior. The high visit frequencies suggest that random walks frequently traverse these nodes, highlighting their role as conduits for information flow, while elevated PageRank scores reinforce their prominence in the network's connectivity.

Furthermore, the network's robustness was evaluated through simulations of central node removal. These assessments revealed specific vulnerabilities, emphasizing the potential impact of targeting these nodes for therapeutic interventions. For example, the removal of *p53-p* resulted in a substantial decrease in overall network connectivity, underscoring its crucial role.

In summary, this study delineates the intricate relationships and trends within the ovarian cancer network, identifying central nodes such as *p53-p*, NF-κB, and *ATM* as vital for maintaining the network's structure and functionality. The network's modular characteristics, recurrent motifs, and the significance of central nodes as demonstrated by random walk and PageRank analyses collectively provide a comprehensive understanding of network dynamics. These insights are instrumental in identifying potential therapeutic targets and elucidating the underlying mechanisms of ovarian cancer progression.

5.1.2 Expectations and Mechanisms

The patterns identified within the ovarian cancer network align with the established biological functions of key nodes. The central role of *p53-p* in this network reflects its well-documented involvement in regulating apoptosis and cell cycle checkpoints. As a critical tumor suppressor gene, *p53-p* orchestrates responses to cellular stress, including mechanisms for inducing cell cycle arrest, promoting DNA repair, and initiating apoptosis in cases of irreparable damage. Its prominence in the network highlights its fundamental importance in preserving cellular integrity and preventing oncogenesis.

Moreover, the central positions of NF-κB and *ATM* in the network underscore their vital functions in inflammation and the DNA damage response, respectively. NF-κB serves as a pivotal regulator of immune response and inflammation, possessing dual roles in promoting cell survival and proliferation while also facilitating apoptotic pathways under specific conditions. Its high centrality suggests a significant impact on modulating the tumor microenvironment, influencing processes such as immune evasion and cancer progression. *ATM* plays an essential role in the DNA damage response by coordinating a network of downstream effectors that facilitate DNA repair and cell cycle regulation, thereby ensuring genomic stability. The centrality of *ATM* in the ovarian cancer network reflects its crucial function in managing genotoxic stress and preserving the integrity of the genome.

The underlying mechanisms of these centrality patterns are elucidated through the integration of key regulatory pathways. *p53-p* serves as an integrative node, responding to signals related to cellular stress and DNA damage and facilitating a coordinated response to maintain cellular homeostasis. Its substantial in-degree and out-degree values reveal extensive interactions with other components of the network, emphasizing its integrative role in stress response mechanisms. NF-κB modulates various pathways in the tumor microenvironment, influencing survival and proliferation while affecting immune responses, further underscoring its significance in cancer biology. *ATM* governs the DNA damage response, activating cascades of proteins involved in repair and cell cycle control, which are essential for maintaining genomic integrity.

The integration of random walk modeling and PageRank analysis provides further insights into the roles of these nodes in network connectivity and stability. The identification of nodes like *Wip1* and *Apoptosis*, which demonstrate high visit frequencies and elevated PageRank scores, underscores their critical functions within the network. The random walk model effectively simulates signal transduction pathways, revealing nodes that frequently facilitate these processes. High PageRank scores indicate the prominence of these nodes, reinforcing their role in ensuring robust information flow and network stability.

Boolean modeling and iterative updates also enrich our understanding of the dynamics within the network. Simulations demonstrated the capacity to model complex biological interactions, offering insights into the regulatory mechanisms governing network stability. Illustrative figures such as **??** and **??** capture the dynamic behavior of the network across various iterations, showcasing the iterative nature of responses to regulatory changes.

Moreover, the training of models using parameters such as epochs, batch size, and accuracy metrics has yielded valuable perspectives on node classification strategies. Improvements in training accuracy and reductions in loss over epochs reflect effective learning in node classification, while significant decreases in training loss indicate successful model refinement. These insights reinforce the model's robustness in capturing complex interactions.

To enhance future research, recommendations include validating models on independent testing sets to ensure generalizability, optimizing hyperparameters for improved performance, and refining data quality and preprocessing methods. Further investigation of misclassified nodes, along with the incorporation of additional graph-based features or ensemble approaches, could enhance model accuracy and interpretability.

In conclusion, this study delineates the intricate molecular mechanisms underlying ovarian cancer by elucidating the roles of central nodes such as *p53-p*, NF-κB, and *ATM*. The utilization of diverse analytical methods advances our understanding of the dynamics within the ovarian cancer network. These insights are pivotal for identifying potential therapeutic targets and for the development of targeted interventions aimed at improving clinical outcomes in ovarian cancer.

5.1.3 Agreement with Previous Work

The results of this study align with existing literature that emphasizes the crucial roles of *p53*, NF-κB, and *ATM* in the context of cancer biology. Previous research has consistently highlighted the significance of these molecules in ovarian cancer, and our analysis further corroborates and expands upon these findings by offering a detailed examination of their interactions and central positions within the signaling network.

p53 is a well-established tumor suppressor gene, recognized for its essential functions in cell cycle regulation and apoptosis. Prior studies have shown that mutations in *p53* frequently occur across various cancer types, including ovarian cancer, leading to impaired cellular responses to DNA damage and promoting tumorigenesis [123, 124]. Our findings reinforce these conclusions by confirming the elevated centrality of *p53* within the ovarian cancer network, underscoring its vital role in maintaining cellular integrity and thwarting malignant progression.

Similarly, NF-κB has been extensively documented for its dual role in inflammation and immune response regulation. Research indicates that NF-κB not only fosters tumor cell survival and proliferation but also influences inflammatory pathways that are critical in the tumor microenvironment [125, 126]. The findings of our study, which position NF-κB prominently within the ovarian cancer network, further validate these earlier assertions and illustrate its significant role in modulating cancer progression.

The *ATM* (Ataxia-Telangiectasia Mutated) kinase, which is pivotal for the DNA damage response and repair, has also been implicated in ovarian cancer through its essential function in preserving genomic stability [127, 128]. Our results corroborate earlier research by demonstrating the centrality of *ATM* within the network, confirming its critical role in orchestrating DNA repair mechanisms and regulating cell cycle checkpoints. This functionality is vital for sustaining genomic integrity, emphasizing *ATM* as an important factor in ovarian cancer biology.

In addition to confirming previous findings, this study employs advanced methodologies such as motif analysis, Boolean modeling, and PageRank algorithms, which contribute to a deeper understanding of the ovarian cancer network.

Motif analysis has enabled the identification of recurring subgraphs and interaction patterns that reveal essential regulatory mechanisms within the network. These motifs, particularly those involving nodes like *TNFa* and *IKKKa*, highlight common themes in the network's structure and functionality [129].

Furthermore, Boolean modeling has facilitated the simulation of the network's dynamic behavior, effectively capturing complex regulatory interactions. The iterative updates and visualizations produced through Boolean simulations have provided valuable insights into how the network responds to perturbations, elucidating the regulatory mechanisms that underpin network stability [120].

The application of PageRank algorithms has also contributed significantly to understanding node importance within the network. The resulting PageRank scores have identified *Wip1* and *Apoptosis* as critical nodes for maintaining network functionality, highlighting their roles in network resilience [130]. This method enhances our comprehension of the hierarchical structure within the network, pinpointing key nodes central to its dynamics.

By integrating these advanced analytical techniques, this study provides a more comprehensive view of the roles of *p53*, NF-κB, and *ATM* in ovarian cancer. The results not only reaffirm previous findings but also elucidate the intricate interactions and regulatory mechanisms involving these pivotal molecules, offering critical insights for potential therapeutic targets and strategies in cancer treatment.

5.1.4 Interpretation in Context of Research Objectives

The primary objective of this study was to develop and implement sophisticated methodologies for identifying critical variables within complex mathematical models of biological systems. This objective was realized through a comprehensive analysis of the ovarian cancer network, where we meticulously examined central nodes and overall stability to address pivotal research questions concerning the regulatory components and their roles in ovarian cancer pathology.

Methodological Framework

Our approach began with the design of a Python pipeline dedicated to data extraction, which facilitated the systematic collection of relevant data files. Following this, we constructed a custom database to store the extracted data, ensuring efficient data management and accessibility. This database allowed for the filtering and arrangement of data in a structured format, which was essential for subsequent visualization and analysis in Cytoscape. Finally, we transferred the organized data back to Python to apply various algorithms for a thorough analysis of the network.

Central Nodes

The results of our study contribute significantly to the field of biological modeling by highlighting critical variables and interaction patterns within the ovarian cancer network. The identification of central nodes such as *p53-p*, NF-κB, and *ATM* exemplifies the efficacy of our methodology in uncovering essential components that govern network dynamics. These central nodes reveal underlying themes in the network's architecture, providing deeper insights into the interactions that sustain the network's integrity and functionality.

Analytical Techniques and Their Impact

The implementation of advanced analytical techniques, including random walk models, PageRank analysis, and Boolean simulations, significantly enhances our understanding of the ovarian cancer network. The random walk model elucidates the pathways through which signals traverse the network, revealing nodes that frequently act as intermediaries in signal transduction processes. This model underscores the importance of nodes like *p53-p*, which, due to its high centrality, plays a pivotal role in orchestrating cellular responses to stress and damage.

PageRank analysis further refines our understanding by quantifying the prominence and connectivity of network nodes. The elevated PageRank scores of nodes such as *Wip1* and *Apoptosis* highlight their critical roles in maintaining network stability and functionality. These findings illustrate the hierarchical structure of the network and identify key nodes integral to its dynamics.

Boolean simulations provide a robust framework for capturing the dynamic behavior of the network under various perturbations. The iterative updates and visualizations generated from these simulations offer insights into the regulatory mechanisms that govern network stability and response to changes. This approach has been instrumental in simulating the effects of perturbations on the network and elucidating the complex interactions among network components.

Implications for Therapeutic Intervention

The evidence presented by these analyses not only confirms the significance of central nodes such as *p53-p*, NF-κB, and *ATM* but also deepens our understanding of their specific roles in ovarian cancer. The elucidation of interaction patterns and network dynamics suggests potential targets for therapeutic intervention. For instance, the central role of *p53-p* in regulating apoptosis and cell cycle checkpoints highlights its potential as a target for therapies aimed at restoring normal cellular responses. Similarly, the prominent position of NF-κB in modulating the tumor microenvironment and its involvement in inflammatory pathways indicate that targeting its signaling pathways could offer novel strategies for combating ovarian cancer. Furthermore, the centrality of *ATM* in DNA damage response underscores its importance as a therapeutic target for enhancing the effectiveness of DNA-damaging agents used in cancer treatment.

Podsumowanie

Niniejsza praca doktorska opracowała i zweryfikowała kompleksowe podejście obliczeniowe do systematycznej identyfikacji kluczowych elementów regulacyjnych w złożonych sieciach biologicznych, ze szczególnym uwzględnieniem raka jajnika. Poprzez integrację danych multi-omics z uzupełniającymi technikami obliczeniowymi - w tym modelowaniem sieci boolowskich, analizą PageRank, metodą losowych spacerów oraz rekurencyjnymi sieciami neuronowymi (RCNN) - badanie to posuwa naprzód biologię systemową i dostarcza praktycznych wniosków dla onkologii precyzyjnej.

Hipoteza 1 została w pełni potwierdzona. We wszystkich zastosowanych warstwach obliczeniowych kanoniczne czynniki regulacyjne, takie jak NF-κB, p53 i ATM, konsekwentnie wyłaniały się jako węzły centralne, co potwierdzono za pomocą miar topologicznych sieci i symulacji dynamicznych. Zbieżność wyników metod boolowskich, analizy PageRank i częstości odwiedzin w losowych spacerach podkreśliła biologiczne znaczenie tych węzłów, zgodnie z hipotezą (Rozdziały 4.9-4.12, 4.15, 5).

Hipoteza 2 również znalazła potwierdzenie. Integracja danych multi-omics z analizą centralności sieciowej umożliwiła identyfikację słabo zbadanych elementów regulacyjnych, takich jak IKKα i Wip1. Węzły te wykazywały wysoką centralność, znaczącą częstość odwiedzin w modelach stochastycznych i związek z potencjalnym znaczeniem funkcjonalnym w sieciach raka jajnika, co uzasadnia ich priorytetową walidację eksperymentalną (Rozdziały 4.12.12, 4.12.13, 5).

Hipoteza 3 została potwierdzona dzięki rygorystycznym testom odpornościowym i sprawdzianowi uogólnialności na różne sieci. Zastosowanie potoku analitycznego do dodatkowych kontekstów biologicznych, w tym do szlaku cyklu komórkowego i szlaku MAPK, konsekwentnie uwidaczniało kluczowe regulatory, takie jak cyklina D, EGFR i ERK. Ponadto, sieci wykazały funkcjonalną odporność, zachowując co najmniej 85% łączności przy losowych perturbacjach węzłów, potwierdzając adaptacyjność frameworka (Rozdziały 4.14.4-4.14.5, 4.11.9).

Poza weryfikacją hipotez, praca ta wnosi modularny, reprodukowalny potok analityczny oparty na Pythonie, zdolny do skalowania na różne systemy biologiczne. Metodologia integruje komponenty grafowo-teoretyczne, dynamiczne i uczenia maszynowego, dostarczając uniwersalnej platformy do identyfikacji kluczowych regulatorów, generowania hipotez i wspierania odkrywania celów terapeutycznych.

Kierunki przyszłych badań obejmują eksperymentalną walidację nowo zidentyfikowanych węzłów (np. Wip1 i IKKα) przy użyciu technologii CRISPR/Cas9 i RNAi, integrację danych single-cell i temporalnych omics dla lepszej rozdzielczości analizy dynamiki sygnałów oraz rozszerzenie na inne modele patologiczne. Połączenie tego potoku z symulacjami farmakokinetycznymi i farmakodynamicznymi otwiera drogę do modelowania predykcyjnego interwencji terapeutycznych.

Podsumowując, niniejsza praca dostarcza rygorystycznego, opartego na danych podejścia łączącego modelowanie obliczeniowe z translacyjnymi badaniami nad nowotworami. Zweryfikowany framework umożliwia systematyczną priorytetyzację kluczowych elementów regulacyjnych w złożonych sieciach biologicznych, stanowiąc podstawę dla bardziej ukierunkowanych i skutecznych strategii terapeutycznych w medycynie precyzyjnej.

Bibliography

[1] Stefan Graw, Kevin Chappell, Charity L Washam, Allen Gies, Jordan Bird, Michael S Robeson, and Stephanie D Byrum. Multi-omics data integration considerations and study design for biological systems and disease. *Molecular omics*, 17(2):170–185, 2021.

[2] OH Houghton, S Mizielinska, and P Gomez-Suaga. The interplay between autophagy and rna homeostasis: implications for amyotrophic lateral sclerosis and frontotemporal dementia. *Frontiers in Cell and Developmental Biology*, 10:838402, 2022.

[3] Douglas Hanahan and Robert A Weinberg. Hallmarks of cancer: the next generation. *cell*, 144(5):646–674, 2011.

[4] Richard C Wang and Zhixiang Wang. Precision medicine: disease subtyping and tailored treatment. *Cancers*, 15(15):3837, 2023.

[5] Genlin Zhang, Feng Qi, Haiyang Jia, Changling Zou, and Chun Li. Advances in bioprocessing for efficient bio manufacture. *Rsc advances*, 5(65):52444–52451, 2015.

[6] Igor Rodchenkov, Ozgun Babur, Augustin Luna, Bulent Arman Aksoy, Jeffrey V Wong, Dylan Fong, Max Franz, Metin Can Siper, Manfred Cheung, Michael Wrana, et al. Pathway commons: 2019 update. *bioRxiv*, page 788521, 2019.

[7] PK Jha and S Ghorai. Stability of prey-predator model with holling type response function and selective harvesting. *Journal of Applied & Computational Mathematics*, 6(3):1–7, 2017.

[8] Uditi Chandrashekhar. The hodgkin-huxley model for neuron action potentials: A computational study. *Macalester Journal of Physics and Astronomy*, 12(1):4, 2024.

[9] Pablo Meyer and Julio Saez-Rodriguez. Advances in systems biology modeling: 10 years of crowdsourcing dream challenges. *Cell Systems*, 12(6):636–653, 2021.

[10] Bin Zhang, Sushil Dhital, and Michael J Gidley. Densely packed matrices as rate determining features in starch hydrolysis. *Trends in Food Science & Technology*, 43(1):18–31, 2015.

[11] Daniel J Levitas and Thomas W James. Dynamic threat–reward neural processing under semi-naturalistic ecologically relevant scenarios. *Human Brain Mapping*, 45(4):e26648, 2024.

[12] Anthony Trewavas. A brief history of systems biology: "every object that biology studies is a system of systems." francois jacob (1974). *The Plant Cell*, 18(10):2420–2430, 2006.

[13] Hiroaki Kitano. Computational systems biology. *Nature*, 420(6912):206–210, 2002.

[14] Andrea Angarita-Rodríguez, Yeimy González-Giraldo, Juan J Rubio-Mesa, Andrés Felipe Aristizábal, Andrés Pinzón, and Janneth González. Control theory and systems biology: Potential applications in neurodegeneration and search for therapeutic targets. *International Journal of Molecular Sciences*, 25(1):365, 2023.

[15] Hana El-Samad. Biological feedback control—respect the loops. *Cell Systems*, 12(6):477–487, 2021.

[16] Xiaofeng Dai and Li Shen. Advances and trends in omics technology development. *Frontiers in Medicine*, 9:911861, 2022.

[17] Heena Satam, Kandarp Joshi, Upasana Mangrolia, Sanober Waghoo, Gulnaz Zaidi, Shravani Rawool, Ritesh P Thakare, Shahid Banday, Alok K Mishra, Gautam Das, et al. Next-generation sequencing technology: current trends and advancements. *Biology*, 12(7):997, 2023.

[18] Kyriacos Felekkis and Christos Papaneophytou. The circulating biomarkers league: Combining mirnas with cell-free dnas and proteins. *International Journal of Molecular Sciences*, 25(6):3403, 2024.

[19] Partho Sen and Matej Orešič. Integrating omics data in genome-scale metabolic modeling: A methodological perspective for precision medicine. *Metabolites*, 13(7):855, 2023.

[20] Jukka Westermarck, Johanna Ivaska, and Garry L Corthals. Identification of protein interactions involved in cellular signaling. *Molecular & Cellular Proteomics*, 12(7):1752–1763, 2013.

[21] Zeyaul Islam, Ameena Mohamed Ali, Adviti Naik, Mohamed Eldaw, Julie Decock, and Prasanna R Kolatkar. Transcription factors: The fulcrum between cell development and carcinogenesis. *Frontiers in oncology*, 11:681377, 2021.

[22] Arathi Nair, Prashant Chauhan, Bhaskar Saha, and Katharina F Kubatzky. Conceptual evolution of cell signaling. *International journal of molecular sciences*, 20(13):3292, 2019.

[23] Hans Peter Fischer. Mathematical modeling of complex biological systems: from parts lists to understanding systems behavior. *Alcohol Research & Health*, 31(1):49, 2008.

[24] Marinka Zitnik, Francis Nguyen, Bo Wang, Jure Leskovec, Anna Goldenberg, and Michael M Hoffman. Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities. *Information Fusion*, 50:71–91, 2019.

[25] Risto Miikkulainen and Stephanie Forrest. A biological perspective on evolutionary computation. *Nature Machine Intelligence*, 3(1):9–15, 2021.

[26] David R Penas, Julio R Banga, Patricia González, and Ramon Doallo. Enhanced parallel differential evolution algorithm for problems in computational systems biology. *Applied Soft Computing*, 33:86–99, 2015.

[27] Peter A Jones, Jean-Pierre J Issa, and Stephen Baylin. Targeting the cancer epigenome for therapy. *Nature Reviews Genetics*, 17(10):630–641, 2016.

[28] Steven L Peck. Simulation as experiment: a philosophical reassessment for biological modeling. *Trends in ecology & evolution*, 19(10):530–534, 2004.

[29] Richard A Erickson, Michael N Fienen, S Grace McCalla, Emily L Weiser, Melvin L Bower, Jonathan M Knudson, and Greg Thain. Wrangling distributed computing for high-throughput environmental science: An introduction to htcondor. *PLoS computational biology*, 14(10):e1006468, 2018.

[30] Wiesława Widłak. High-throughput technologies in molecular biology. In *Molecular biology: not only for bioinformaticians*, pages 139–153. Springer, 2013.

[31] Bharat Mishra, Nilesh Kumar, and M Shahid Mukhtar. Systems biology and machine learning in plant–pathogen interactions. *Molecular Plant-Microbe Interactions*, 32(1):45–55, 2019.

[32] S Das and PK Gupta. A mathematical model on fractional lotka–volterra equations. *Journal of theoretical biology*, 277(1):1–6, 2011.

[33] Kai Diethelm and Neville J Ford. Analysis of fractional differential equations. *Journal of Mathematical Analysis and Applications*, 265(2):229–248, 2002.

[34] Baoshan Ma, Mingkun Fang, and Xiangtian Jiao. Inference of gene regulatory networks based on nonlinear ordinary differential equations. *Bioinformatics*, 36(19):4885–4893, 2020.

[35] Andre S Ribeiro. Stochastic and delayed stochastic models of gene expression and regulation. *Mathematical biosciences*, 223(1):1–11, 2010.

[36] Birgit Wiltschi, Tomislav Cernava, Alexander Dennig, Meritxell Galindo Casas, Martina Geier, Steffen Gruber, Marianne Haberbauer, Petra Heidinger, Enrique Herrero Acero, Regina Kratzer, et al. Enzymes revolutionize the bioproduction of value-added compounds: From enzyme discovery to special applications. *Biotechnology advances*, 40:107520, 2020.

[37] Boseung Choi, Grzegorz A Rempala, and Jae Kyoung Kim. Beyond the michaelis-menten equation: Accurate and efficient estimation of enzyme kinetic parameters. *Scientific reports*, 7(1):17018, 2017.

[38] Hafeez Aderinsayo Adekola, Ibrahim Ayoade Adekunle, Haneefat Olabimpe Egberongbe, Sefiu Adekunle Onitilo, and Idris Nasir Abdullahi. Mathematical modeling for infectious viral disease: The covid-19 perspective. *Journal of public affairs*, 20(4):e2306, 2020.

[39] Shweta Bansal, Bryan T Grenfell, and Lauren Ancel Meyers. When individual behaviour matters: homogeneous and network models in epidemiology. *Journal of the Royal Society Interface*, 4(16):879–891, 2007.

[40] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

[41] Chaimaa Tarzi, Guido Zampieri, Neil Sullivan, and Claudio Angione. Emerging methods for genome-scale metabolic modeling of microbial communities. *Trends in Endocrinology & Metabolism*, 2024.

[42] Friedemann Pulvermüller, Rosario Tomasello, Malte R Henningsen-Schomers, and Thomas Wennekers. Biological constraints on neural network models of cognitive function. *Nature Reviews Neuroscience*, 22(8):488–502, 2021.

[43] Huixi Zou, Parikshit Banerjee, Sharon S. Y. Leung, and Xiaoyu Yan. Application of pharmacokinetic-pharmacodynamic modeling in drug delivery: development and challenges. *Frontiers in Pharmacology*, 11:997, 2020.

[44] Allen P Minton. The influence of macromolecular crowding and macromolecular confinement on biochemical reactions in physiological media. *Journal of biological chemistry*, 276(14):10577–10580, 2001.

[45] T Sumner, E Shephard, and IDL Bogle. A methodology for global-sensitivity analysis of time-dependent outputs in systems biology modelling. *Journal of The Royal Society Interface*, 9(74):2156–2166, 2012.

[46] James M Gallo, Paolo Vicini, Amy Orlansky, Shaolan Li, Feng Zhou, Jianguo Ma, Sharon Pulfer, Michel A Bookman, and Ping Guo. Pharmacokinetic model-predicted anticancer drug concentrations in human tumors. *Clinical cancer research*, 10(23):8048–8058, 2004.

[47] Mojtaba Barzegari, Di Mei, Sviatlana V. Lamaka, and Liesbet Geris. Computational modeling of degradation process of biodegradable magnesium biomaterials. *Corrosion Science*, 190:109674, 2021.

[48] Martin Garrido-Rodriguez, Katharina Zirngibl, Olga Ivanova, Sebastian Lobentanzer, and Julio Saez-Rodriguez. Integrating knowledge and omics to decipher mechanisms via large-scale models of signaling networks. *Molecular systems biology*, 18(7):e11036, 2022.

[49] S Bek, JV Nielsen, Anders Bo Bojesen, A Franke, Steffen Bank, U Vogel, and V Andersen. Systematic review: genetic biomarkers associated with anti-tnf treatment response in inflammatory bowel diseases. *Alimentary pharmacology & therapeutics*, 44(6):554–567, 2016.

[50] Lisette G de Pillis, Ami E Radunskaya, and Charles L Wiseman. A validated mathematical model of cell-mediated immune response to tumor growth. *Cancer research*, 65(17):7950–7958, 2005.

[51] Mohammadreza Mohaghegh Neyshabouri, Seong-Hwan Jun, and Jens Lagergren. Inferring tumor progression in large datasets. *PLoS computational biology*, 16(10):e1008183, 2020.

[52] Evelyn Trummer, Katharina Fauland, Silke Seidinger, Kornelia Schriebl, Christine Lattenmayer, Renate Kunert, Karola Vorauer-Uhl, Robert Weik, Nicole Borth, Hermann Katinger, et al. Process parameter shifting: Part ii. biphasic cultivation—a tool for enhancing the volumetric productivity of batch processes using epo-fc expressing cho cells. *Biotechnology and bioengineering*, 94(6):1045–1052, 2006.

[53] Pramod R Somvanshi, Anilkumar K Patel, Sharad Bhartiya, and KV Venkatesh. Implementation of integral feedback control in biological systems. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 7(5):301–316, 2015.

[54] Jin Hwan Park, Sang Yup Lee, Tae Yong Kim, and Hyun Uk Kim. Application of systems biology for bioprocess development. *Trends in biotechnology*, 26(8):404–412, 2008.

[55] DK Arrell and Andre Terzic. Network systems biology for drug discovery. *Clinical Pharmacology & Therapeutics*, 88(1):120–125, 2010.

[56] Matthew Freeman. Feedback control of intercellular signalling in development. *Nature*, 408(6810):313–319, 2000.

[57] Nathaniel J Linden, Boris Kramer, and Padmini Rangamani. Bayesian parameter estimation for dynamical models in systems biology. *PLoS computational biology*, 18(10):e1010651, 2022.

[58] Claudine Chaouiya, Sarah M Keating, Duncan Berenguier, Aurélien Naldi, Denis Thieffry, Martijn P van Iersel, Nicolas Le Novere, and Tomáš Helikar. Sbml level 3 package: qualitative models, version 1, release 1. *Journal of integrative bioinformatics*, 12(2):691–730, 2015.

[59] María Concepción Bielza Lozoya, Serafín Moral Callejón, and Antonio Salmerón Cerdán. Recent advances in probabilistic graphical models. *International Journal of Intelligent Systems*, 30(3):207–208, 2015.

[60] Giulia Tini, Luca Marchetti, Corrado Priami, and Marie-Pier Scott-Boyer. Multi-omics integration—a comparison of unsupervised clustering methodologies. *Briefings in bioinformatics*, 20(4):1269–1279, 2019.

[61] Shuhui Bi, Mingcong Deng, and Yongfei Xiao. Robust stability and tracking for operator-based nonlinear uncertain systems. *IEEE Transactions on Automation Science and Engineering*, 12(3):1059–1066, 2014.

[62] A Kremling and J Saez-Rodriguez. Systems biology—an engineering perspective. *Journal of biotechnology*, 129(2):329–351, 2007.

[63] Claire J Tomlin and Jeffrey D Axelrod. Biology by numbers: mathematical modelling in developmental biology. *Nature reviews genetics*, 8(5):331–340, 2007.

[64] Eric Mjolsness. Prospects for declarative mathematical modeling of complex biological systems. *Bulletin of Mathematical Biology*, 81(8):3385–3420, 2019.

[65] G Wayne Brodland. How computational models can help unlock biological systems. In *Seminars in cell & developmental biology*, volume 47, pages 62–73. Elsevier, 2015.

[66] Rachel Walker and Heiko Enderling. From concept to clinic: mathematically informed immunotherapy. *Current Problems in Cancer*, 40(1):68–83, 2016.

[67] Otto Richter. Modelling dispersal of populations and genetic information by finite element methods. *Environmental Modelling & Software*, 23(2):206–214, 2008.

[68] Qiao Zhuang and Jin Wang. A spatial epidemic model with a moving boundary. *Infectious Disease Modelling*, 6:1046–1060, 2021.

[69] Florian Hartig, Justin M Calabrese, Björn Reineking, Thorsten Wiegand, and Andreas Huth. Statistical inference for stochastic simulation models–theory and application. *Ecology letters*, 14(8):816–827, 2011.

[70] Wesley Tansey, Christopher Tosh, and David M Blei. A bayesian model of dose-response for cancer drug studies. *arXiv preprint arXiv:1906.04072*, 2019.

[71] Vassily Hatzimanikatis, Christodoulos A Floudas, and James E Bailey. Optimization of regulatory architectures in metabolic reaction networks. *Biotechnology and bioengineering*, 52(4):485–500, 1996.

[72] ÖMÜR Uğur, SW Pickl, G-W Weber, and R Wünschiers. An algorithmic approach to analyse genetic networks and biological energy production: an introduction and contribution where or meets biology. *Optimization*, 58(1):1–22, 2009.

[73] Fawang Liu and Kevin Burrage. Novel techniques in parameter estimation for fractional dynamical models arising from biological systems. *Computers & Mathematics with Applications*, 62(3):822–833, 2011.

[74] Hubert C Chua and Vincent H Tam. Optimizing clinical outcomes through rational dosing strategies: Roles of pharmacokinetic/pharmacodynamic modeling tools. In *Open Forum Infectious Diseases*, volume 9, page ofac626. Oxford University Press US, 2022.

[75] Wei Gao, Hualong Wu, M. K. Siddiqui, and A. Q. Baig. Study of biological networks using graph theory. *Saudi Journal of Biological Sciences*, 25:1212 – 1219, 2017.

[76] A. Amara, Clément Frainay, F. Jourdan, Thomas Naake, S. Neumann, E. Novoa del Toro, R. Salek, Liesa Salzer, Sarah Scharfenberg, and M. Witting. Networks and graphs discovery in metabolomics data analysis and interpretation. *Frontiers in Molecular Biosciences*, 9, 2022.

[77] Oana-Teodora Chis, Julio R Banga, and Eva Balsa-Canto. Structural identifiability of systems biology models: a critical comparison of methods. *PloS one*, 6(11):e27755, 2011.

[78] Benjamin D Weger, Cédric Gobet, Fabrice PA David, Florian Atger, Eva Martin, Nicholas E Phillips, Aline Charpagne, Meltem Weger, Felix Naef, and Frédéric Gachon. Systematic analysis of differential rhythmic liver gene expression mediated by the circadian clock and feeding rhythms. *Proceedings of the National Academy of Sciences*, 118(3):e2015803118, 2021.

[79] Diogo M Camacho, Katherine M Collins, Rani K Powers, James C Costello, and James J Collins. Next-generation machine learning for biological networks. *Cell*, 173(7):1581–1592, 2018.

[80] Qi Song, Jiyoung Lee, Shamima Akter, Matthew Rogers, Ruth Grene, and Song Li. Prediction of condition-specific regulatory genes using machine learning. *Nucleic Acids Research*, 48(11):e62–e62, 2020.

[81] Laura Keller and Klaus Pantel. Unravelling tumour heterogeneity by single-cell profiling of circulating tumour cells. *Nature Reviews Cancer*, 19(10):553–567, 2019.

[82] Florian Rohart, Benoît Gautier, Amrit Singh, and Kim-Anh Lê Cao. mixomics: An r package for ‘omics feature selection and multiple data integration. *PLoS computational biology*, 13(11):e1005752, 2017.

[83] Fulvia Ferrazzi, Paola Sebastiani, Marco F Ramoni, and Riccardo Bellazzi. Bayesian approaches to reverse engineer cellular systems: a simulation study on nonlinear gaussian networks. *BMC bioinformatics*, 8:1–15, 2007.

[84] Mufti Mahmud, M Shamim Kaiser, T Martin McGinnity, and Amir Hussain. Deep learning in mining biological data. *Cognitive computation*, 13(1):1–33, 2021.

[85] Julia M Rohrer. Thinking clearly about correlations and causation: Graphical causal models for observational data. *Advances in methods and practices in psychological science*, 1(1):27–42, 2018.

[86] Hyo Jung Kang, Yoon Sik Choi, Seung-Beom Hong, Kee-Won Kim, Ran-Sook Woo, Seok Joon Won, Eun Ju Kim, Hee Kyung Jeon, So-Young Jo, Tae Kook Kim, et al. Ectopic expression of the catalytic subunit of telomerase protects against brain injury resulting from ischemia and nmda-induced neurotoxicity. *Journal of Neuroscience*, 24(6):1280–1287, 2004.

[87] Viet-Thi Tran and Philippe Ravaud. Collaborative open platform e-cohorts for research acceleration in trials and epidemiology. *Journal of clinical epidemiology*, 124:139–148, 2020.

[88] Y Swathi and Manoj Challa. A comparative analysis of explainable ai techniques for enhanced model interpretability. In *2023 3rd International Conference on Pervasive Computing and Social Networking (ICPCSN)*, pages 229–234. IEEE, 2023.

[89] Zuan-Fu Lim and Patrick C Ma. Emerging insights of tumor heterogeneity and drug resistance mechanisms in lung cancer targeted therapy. *Journal of hematology & oncology*, 12(1):134, 2019.

[90] Ildikó Csóka, Edina Pallagi, and Tamás L Paál. Extension of quality-by-design concept to the early development phase of pharmaceutical r&d processes. *Drug Discovery Today*, 23(7):1340–1343, 2018.

[91] Zhou Li, Song Cui, Qingping Zhang, Gang Xu, Qisheng Feng, Chao Chen, and Yuan Li. Optimizing wheat yield, water, and nitrogen use efficiency with water and nitrogen inputs in china: A synthesis and life cycle assessment. *Frontiers in Plant Science*, 13:930484, 2022.

[92] Lindsay Wessel, Yi Hua, Jianhong Wu, and Seyed M Moghadas. Public health interventions for epidemics: implications for multiple infection waves. *BMC Public Health*, 11:1–9, 2011.

[93] Christopher P Weaver, Robert J Lempert, Casey Brown, John A Hall, David Revell, and Daniel Sarewitz. Improving the contribution of climate model information to decision making: the value

and demands of robust decision frameworks. *Wiley Interdisciplinary Reviews: Climate Change*, 4(1):39–60, 2013.

[94] Timothy K Lu, Ahmad S Khalil, and James J Collins. Next-generation synthetic gene networks. *Nature biotechnology*, 27(12):1139–1150, 2009.

[95] Per Hage and Frank Harary. Eccentricity and centrality in networks. *Social networks*, 17(1):57–63, 1995.

[96] Hassan Sayyadi and Lise Getoor. Futurerank: ranking scientific articles by predicting their future pagerank. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pages 533–544. SIAM, 2009.

[97] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. 1999.

[98] Mark EJ Newman. A measure of betweenness centrality based on random walks. *Social networks*, 27(1):39–54, 2005.

[99] Arunachalam Vinayagam, Travis E Gibson, Ho-Joon Lee, Bahar Yilmazel, Charles Roesel, Yanhui Hu, Young Kwon, Amitabh Sharma, Yang-Yu Liu, Norbert Perrimon, et al. Controllability analysis of the directed human protein interaction network identifies disease genes and drug targets. *Proceedings of the National Academy of Sciences*, 113(18):4976–4981, 2016.

[100] Javier Sanz-Cruzado and Pablo Castells. Relison: a framework for link recommendation in social networks. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2992–3002, 2022.

[101] Roland Molontay and Marcell Nagy. Two decades of network science: as seen through the co-authorship network of network scientists. In *Proceedings of the 2019 IEEE/ACM international conference on advances in social networks analysis and mining*, pages 578–583, 2019.

[102] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, 1977.

[103] Mark E. J. Newman. *Networks: An Introduction.* Oxford University Press, 2018.

[104] Béla Bollobás. Random graphs. In *Modern graph theory*, pages 215–252. Springer, 1998.

[105] László Lovász and Miklós Simonovits. Random walks in a convex body and an improved volume algorithm. *Random structures & algorithms*, 4(4):359–412, 1993.

[106] Markus Brede. Networks—an introduction. mark ej newman.(2010, oxford university press.) $65.38, £35.96 (hardcover), 772 pages. isbn-978-0-19-920665-0.$, 2012.

[107] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nature reviews genetics*, 12(1):56–68, 2011.

[108] Jonas Forsman, Maartje Van den Bogaard, Cedric Linder, and Duncan Fraser. Considering student retention as a complex system: a possible way forward for enhancing student retention. *European Journal of Engineering Education*, 40(3):235–255, 2015.

[109] Phillip Bonacich. Power and centrality: A family of measures. *American Journal of Sociology*, 92(5):1170–1182, 1987.

[110] Varun Naganathan. *Network-based and Molecular Dynamics Analysis of Dynamic Allostery in Proteins.* PhD thesis, International Institute of Information Technology, Hyderabad, 2022.

[111] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

[112] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

[113] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[114] M. E. J. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27(1):39–54, 2005.

[115] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[116] Petter Holme and Jari Saramäki. Temporal networks. *Physics Reports*, 519(3):97–125, 2012.

[117] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, 1977.

[118] The Cancer Genome Atlas Research Network. The cancer genome atlas pan-cancer analysis project. *Nature Genetics*, 45(10):1113–1120, 2013.

[119] I. Rodchenkov et al. Pathway commons 2021 update: A resource for pathway-based analysis. *Nucleic Acids Research*, 49(D1):D613–D621, 2021.

[120] Stuart A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437–467, 1969.

[121] Brittney S. Harrington and Christina M. Annunziata. Nf-κb signaling in ovarian cancer. *Cancers*, 11(8):1182, 2019.

[122] Anika Maria Weber and Anderson Joseph Ryan. Atm and atr as therapeutic targets in cancer. *Pharmacology & therapeutics*, 149:124–138, 2015.

[123] David P. Lane and L. V. Crawford. Tumor suppressor genes. *Nature*, 358(6381):15–16, 1992.

[124] Bert Vogelstein and Kenneth W. Kinzler. The cancer genome. *Nature*, 421(6921):579–584, 2000.

[125] Michael Karin and Florian R. Greten. Nf-kb: Linking inflammation and immunity to cancer development and progression. *Nature Reviews Immunology*, 6(5):313–324, 2006.

[126] Neil D. Perkins. Integrating cell-signalling pathways with nf-kb and ikk function. *Nature Reviews Molecular Cell Biology*, 8(6):449–462, 2007.

[127] Angela Ciccia and Stephen J. Elledge. The dna damage response: Making it safe to play with knives. *Molecular Cell*, 40(2):179–204, 2012.

[128] Y. Shiloh. Atm and related protein kinases: Safeguarding genome integrity. *Nature Reviews Cancer*, 3(3):155–168, 2003.

[129] Rotem Milo, Shai S. Shen-Orr, Shai Itzkovitz, Nira Kashtan, et al. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[130] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report Technical Report, Stanford InfoLab, 1999.

.1 Appendix: Approval

.1.1 Final Statement of the Doctoral Candidate

I am aware of the possible disciplinary consequences of providing incorrect information in the application and of violating the rules and regulations in effect at IDS SUT, in accordance with the Act of July 20, 2018, Law on Higher Education and Science (i.e., Journal of Laws of 2020, item 85, as amended). With my authorization, I affirm the accuracy of the data contained in this submission.

PhD Student's Signature	**Supervisor's Signature**
Podpisała Ruby Khan (podpis odręczny)	Podpisał promotor nazwiskiem Puszyński (podpis odręczny)

Appendix A

Supplementary Tables

Python Script

The following Python script downloads and processes PathwayCommons data, formats it into a DataFrame, and saves it as a CSV file.

```
import pandas as pd
import contextlib
import urllib.request
import gzip
from setuptools import setup

# Package setup configuration
setup(
    name=’my_library’,
    version=’0.1’,
    packages=[’my_library’],
    install_requires=[’pandas’]
)

# Function to download and process PathwayCommons data
def DownloadAndProcessPathwayCommonsData(filename="PathwayCommons.csv"):
    url = "http://www.pathwaycommons.org/archives/PC2/v12/PathwayCommons12.All.
        BINARY_SIF.gz"
    output_file = "PathwayCommons.gz"

    # Downloading the file
    urllib.request.urlretrieve(url, output_file)

    # Extract and process data
    with gzip.open(output_file, ’rt’) as f:
        data = [line.strip().split("\t") for line in f.readlines()]

    # Convert data into a pandas DataFrame
    df = pd.DataFrame(data, columns=["Participant A", "Interaction Type", "
        Participant B", "Source", "PubMed ID", "Pathway Names"])

    # Filtering data based on interaction types
    interaction_types = {"controls-state-change-of", "controls-phosphorylation-of",
        "controls-expression-of"}
    df_filtered = df[df["Interaction Type"].isin(interaction_types)]

    # Drop unnecessary columns
    df_filtered = df_filtered[["Participant A", "Participant B", "Interaction Type"
        ]]

    # Save to CSV
    df_filtered.to_csv(filename, index=False)

    return "Data downloaded and processed successfully."

# Main execution
if __name__ == "__main__":
    result = DownloadAndProcessPathwayCommonsData(filename="D://ProjectFiles//
        PathwayCommon.csv")
    print(result)

# Encapsulating in a class
class PathwayDataLoader:
    def __init__(self, filename="PathwayCommons.csv"):
        self.filename = filename

    def download_and_process(self):
        return DownloadAndProcessPathwayCommonsData(self.filename)

# Using the class for data processing
if __name__ == "__main__":
    loader = PathwayDataLoader(filename="D://ProjectFiles//PathwayCommon.csv")
    result = loader.download_and_process()
    print(result)
```

Listing A.1: Listing A.5: Encapsulating in a class

Script Summary

Python Script

```
import pandas as pd
import urllib.request
import gzip
import contextlib

# Function to download and process CellTalk data
def DownloadAndProcessCellTalkData(filename="CellTalk.csv"):
    url = "https://example.com/CellTalkData.gz"
    output_file = "CellTalkData.gz"

    # Downloading the file
    urllib.request.urlretrieve(url, output_file)

    # Extract and process data
    with gzip.open(output_file, 'rt') as f:
        data = [line.strip().split("\t") for line in f.readlines()]

    # Convert data into a pandas DataFrame
    df = pd.DataFrame(data, columns=[
        "lr_pair", "ligand_gene_symbol", "receptor_gene_symbol",
        "ligand_gene_id", "receptor_gene_id", "ligand_ensembl_protein_id",
        "receptor_ensembl_protein_id", "ligand_ensembl_gene_id",
        "receptor_ensembl_gene_id", "evidence"
    ])

    # Save to CSV
    df.to_csv(filename, index=False)

    return "Data downloaded and processed successfully."

# Main execution
if __name__ == "__main__":
    result = DownloadAndProcessCellTalkData(filename="D://ProjectFiles//CellTalk.csv
        ")
    print(result)

# Encapsulating in a class
class CellTalkDataLoader:
    def __init__(self, filename="CellTalk.csv"):
        self.filename = filename

    def download_and_process(self):
        return DownloadAndProcessCellTalkData(self.filename)

# Using the class for data processing
if __name__ == "__main__":
    loader = CellTalkDataLoader(filename="D://ProjectFiles//CellTalk.csv")
    result = loader.download_and_process()
    print(result)
```

Listing A.2: Listing A.6: Processing CellTalk Data

A.0.1 Processing AnimalTF Data

```
import pandas as pd
import urllib.request
import gzip

def DownloadAndProcessAnimalTFData(filename="AnimalTFData.csv"):
    url = "https://example.com/AnimalTFData.gz"
    output_file = "AnimalTFData.gz"

    urllib.request.urlretrieve(url, output_file)

    with gzip.open(output_file, ’rt’) as f:
        data = [line.strip().split("\t") for line in f.readlines()]

    df = pd.DataFrame(data, columns=["Species", "Symbol", "Ensembl", "Family", "
        Protein", "Entrez_ID"])
    df.to_csv(filename, index=False)
    return "AnimalTF data downloaded and processed successfully."
```

Listing A.3: Processing AnimalTF Data

A.0.2 Processing TCGA-OV Data

```
import pandas as pd
import requests
import io

def DownloadAndProcessTCGA_OVData():
    endpoint = "https://api.gdc.cancer.gov/files"
    filters = {
        "filters": {
            "op": "and",
            "content": [
                {"op": "in", "content": {"field": "cases.project.project_id", "value
                    ": ["TCGA-OV"]}},
                {"op": "in", "content": {"field": "files.data_format", "value": ["
                    BAM"]}},
                {"op": "in", "content": {"field": "files.experimental_strategy", "
                    value": ["WGS"]}}
            ]
        },
        "format": "TSV",
        "fields": "file_id,file_name,data_category,data_type,md5sum,file_size",
        "size": "1000"
    }

    response = requests.post(endpoint, json=filters, headers={"Content-Type": "
        application/json"})
    df_wgs = pd.read_csv(io.StringIO(response.text), sep="\t")
    df_wgs = df_wgs[df_wgs["data_type"] == "Aligned Reads"]
    df_wgs.to_csv("TCGA-OV_WGS_files.csv", index=False)
    return "TCGA-OV WGS data downloaded and processed successfully."
```

Listing A.4: Processing TCGA-OV Data

A.0.3 Automated Data Extraction from GDC TCGA-OV

Python Script for Extracting TCGA-OV Ovarian Cancer Data

Below is the Python script used for automatically querying and downloading ovarian cancer data from the Genomic Data Commons (GDC) TCGA-OV project.

```python
import requests
import json
import pandas as pd
import os

# Define the base URL for the GDC API
GDC_API = "https://api.gdc.cancer.gov/"

# Step 1: Define the query for TCGA OV data
def query_gdc(project="TCGA-OV", data_type="Gene Expression Quantification", workflow_type="HTSeq - Counts"):
    filters = {
        "op": "and",
        "content": [
            {
                "op": "in",
                "content": {
                    "field": "cases.project.project_id",
                    "value": [project]
                }
            },
            {
                "op": "in",
                "content": {
                    "field": "files.data_type",
                    "value": [data_type]
                }
            },
            {
                "op": "in",
                "content": {
                    "field": "files.analysis.workflow_type",
                    "value": [workflow_type]
                }
            }
        ]
    }

    params = {
        "filters": json.dumps(filters),
        "format": "json",
        "size": "1000",  # Limit results to 1000 entries
        "fields": "file_id,file_name,cases.case_id,cases.submitter_id,data_type"
    }

    response = requests.get(GDC_API + "files", params=params)

    if response.status_code == 200:
        print("Query successful!")
        return response.json()
    else:
        print(f"Error: {response.status_code}")
        return None

# Step 2: Parse and save the metadata
def save_metadata(metadata, output_filename="gdc_tcga_ov_metadata.csv"):
    data = [
        {
            "file_id": file_entry["file_id"],
            "file_name": file_entry["file_name"],
            "case_id": file_entry["cases"][0]["case_id"],
            "submitter_id": file_entry["cases"][0]["submitter_id"],
            "data_type": file_entry["data_type"]
        }
        for file_entry in metadata["data"]["hits"]
    ]

    df = pd.DataFrame(data)
    df.to_csv(output_filename, index=False)
    print(f"Metadata saved to {output_filename}")
    return df

# Step 3: Download the files based on file IDs
def download_files(file_ids, download_dir="gdc_tcga_ov_data"):
    os.makedirs(download_dir, exist_ok=True)

    for file_id in file_ids:
        download_url = f"{GDC_API}data/{file_id}"
        response = requests.get(download_url, stream=True)
        if response.status_code == 200:
            file_path = os.path.join(download_dir, f"{file_id}.tar.gz")
            with open(file_path, "wb") as f:
                for chunk in response.iter_content(chunk_size=1024):
                    if chunk:
                        f.write(chunk)
            print(f"Downloaded {file_id} to {file_path}")
        else:
            print(f"Failed to download file {file_id}")

# Main function to execute the steps
def main():
    metadata = query_gdc()
    if metadata:
        df_metadata = save_metadata(metadata)
        file_ids = df_metadata["file_id"].tolist()
        download_files(file_ids)

if __name__ == "__main__":
    main()
```

Listing A.5: Python Script for TCGA-OV Data Extraction

```python
import networkx as nx
import matplotlib.pyplot as plt
from matplotlib.lines import Line2D
import numpy as np

# Create the network
G = nx.DiGraph()

# Add edges with interaction types
edges = [
    ("p53", "p53-p", "phosphorylation"),
    ("p53 mRNA", "p53", "translation"),
    # ... (all other edges from your data)
    ("NFkB", "Wip1 mRNA", "transcription")
]

for source, target, interaction in edges:
    G.add_edge(source, target, interaction=interaction)

# Add any isolated nodes
```

```
all_nodes = {
    ’p53’, ’p53-p’, ’p53 mRNA’, # ... (all other nodes)
    ’pre-miR-16’, ’miR-16’
}

for node in all_nodes:
    if node not in G:
        G.add_node(node)
```

Listing A.6: Network construction and basic visualization

```
# Set up the figure with larger size
plt.figure(figsize=(24, 18))

# Define node positions using spring layout with adjustments
pos = nx.spring_layout(G, k=1.5, seed=42, iterations=200, scale=2)

# Define node colors and shapes based on function
node_colors = []
node_shapes = []
for node in G.nodes():
    if ’mRNA’ in node:
        node_colors.append(’#FFA07A’)  # Light salmon
        node_shapes.append(’s’)  # Square
    elif node in [’apoptosis’, ’cell cycle arrest’]:
        # ... (other node coloring rules)
    else:
        node_colors.append(’#D3D3D3’)  # Light gray
        node_shapes.append(’o’)  # Circle

# Draw nodes with different shapes
for shape in set(node_shapes):
    nodes = [node for node, s in zip(G.nodes(), node_shapes) if s == shape]
    nx.draw_networkx_nodes(G, pos, nodelist=nodes,
                          node_shape=shape,
                          node_color=[node_colors[list(G.nodes()).index(node)]
                                    for node in nodes],
                          node_size=2000, alpha=0.9)

# Draw edges with different styles
for i, (u, v, data) in enumerate(G.edges(data=True)):
    nx.draw_networkx_edges(G, pos, edgelist=[(u, v)],
                          width=edge_widths[i],
                          edge_color=edge_colors[i],
                          arrowsize=25,
                          arrowstyle=’->’,
                          connectionstyle=’arc3,rad=0.2’)

# Save high-quality image
plt.savefig(’p53_signaling_network_improved.png’,
           dpi=600, bbox_inches=’tight’, transparent=True)
```

Listing A.7: Enhanced visualization with custom styling

```
# Biochemical Flowchart Layout
def create_biochemical_layout(G):
    pos = {
        # Input signals
        ’IR’: (0, 10), ’TNFa’: (0, 8),
        # DNA damage response
        ’DSB’: (3, 11), ’ATM-p’: (6, 11),
        # ... (other positioned nodes)
    }
    # Position remaining nodes programmatically
    y_pos = 3
    x_pos = 15
    for node in G.nodes():
        if node not in pos:
            pos[node] = (x_pos, y_pos)
            y_pos -= 0.8
            if y_pos < -5:
                y_pos = 3
                x_pos += 3
    return pos

# Molecular Constellation Layout
def create_constellation_layout(G):
    pos = {'p53': (0,0)}
    functional_groups = {
        'DNA Damage': ['DSB', 'ATM-p', ...],
        # ... (other groups)
    }
    # Position functional groups radially
    # ... (positioning code)
    return pos

# Generate and save all visualizations
layouts = {
    'biochemical': create_biochemical_layout,
    'constellation': create_constellation_layout,

}

for name, layout_func in layouts.items():
    plt.figure(figsize=(20, 20))
    pos = layout_func(G)
    # Draw network with this layout

    plt.savefig(f'p53_{name}_layout.png', dpi=300)
```

Listing A.8: Alternative layout visualizations

```
import networkx as nx
import matplotlib.pyplot as plt
from pyvis.network import Network

# [Previous node and interaction definitions...]

# Create matplotlib visualization
plt.figure(figsize=(25, 25))
pos = nx.spring_layout(G, k=0.5, seed=42)  # k controls node spacing

# Color nodes by function
node_colors = []
for node in G.nodes():
    if 'mRNA' in node:
        node_colors.append('orange')
    elif 'p' in node or 'ac' in node or 'ub' in node:
        node_colors.append('lime')
    elif node in ['apoptosis', 'cell cycle arrest']:
        node_colors.append('purple')
    elif 'p53' in node:
        node_colors.append('red')
    else:
        node_colors.append('skyblue')

nx.draw(G, pos, with_labels=True, node_size=1200,
       node_color=node_colors, font_size=8,
       edge_color='gray', width=1.0, arrowsize=20,
       arrowstyle='->')

plt.title("Comprehensive p53 Signaling Network", size=20)
plt.tight_layout()
plt.savefig('p53_signaling_network.png', dpi=300, bbox_inches='tight')
plt.close()

# Create interactive visualization
net = Network(notebook=True, height="900px", width="100%", directed=True, bgcolor='
    #222222')
net.from_nx(G)
```

```

# Customize node appearance
for node in net.nodes:
    if 'mRNA' in node['label']:
        node['color'] = 'orange'
        node['shape'] = 'diamond'
        node['size'] = 20
    elif 'p' in node['label'] or 'ac' in node['label'] or 'ub' in node['label']:
        node['color'] = 'lime'
        node['shape'] = 'triangle'
    elif node['label'] in ['apoptosis', 'cell cycle arrest']:
        node['color'] = 'purple'
        node['shape'] = 'star'
        node['size'] = 30
    elif 'p53' in node['label']:
        node['color'] = 'red'
        node['size'] = 25
    else:
        node['color'] = 'skyblue'

    node['font'] = {'size': 10, 'color': 'white'}

# Customize edge appearance
for edge in net.edges:
    edge['width'] = 1.5
    edge['color'] = '#cccccc'
    edge['title'] = edge['interaction']  # Show interaction type on hover

net.show_buttons(filter_=['physics', 'nodes', 'edges'])
net.save_graph('interactive_p53_network.html')

print("\nVisualizations created:")
print("- Static visualization saved as 'p53_signaling_network.png'")
print("- Interactive visualization saved as 'interactive_p53_network.html'")
```

Listing A.9: Complete Python implementation of p53 network construction, analysis, and visualization

Supplementary Code

```
import os
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, LSTM, Dense, Dropout, GlobalAveragePooling1D
from tensorflow.keras.callbacks import EarlyStopping
import networkx as nx
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report
import shap

# To ensure reproducibility
np.random.seed(42)
tf.random.set_seed(42)

# Define the graph
G = nx.DiGraph()
G.add_nodes_from(
    ['TNFa', 'TNFR1', 'IKKKa', 'IKKa', 'NFkB', 'p53', 'p53-p', 'A20 mRNA', 'A20', 'IkBa mRNA',
    'Mdm2 cyt', 'IkBa', 'Wip mRNA', 'Wip1', 'p53 mRNA', 'ATM mRNA', 'ATM', 'ATM-p', 'ATMa-p', 'MRN-p',
    ’Chk2-p’, ’CREB’, ’KSRP-p’, ’AKT-p’, ’Mdm2-p nuc’, ’PTEN mRNA’, ’PTEN’, ’PIP2’, ’PIP3’, ’Bax mRNA’,
    ’Bax’, ’apoptosis’, ’Mdm2 mRNA’, ’p21 mRNA’, ’p21’, ’cell cycle arrest’, ’Chk2 mRNA’, ’Chk2’, ’IR’, ’DSB’,
    ’miR-16’, ’pre-miR-16’])
G.add_edges_from([
    (’TNFa’, ’TNFR1’), (’TNFR1’, ’IKKKa’), (’IKKKa’, ’IKKa’),
    (’IKKa’, ’NFkB’), (’TNFR1’, ’NFkB’), (’ATMa-p’, ’IKKa’), (’p53’, ’p53-p’),
    (’PIP2’, ’PIP3’), (’p53 mRNA’, ’p53’), (’DSB’, ’ATM-p’), (’p53-p’, ’ATM mRNA’),
    (’ATMa-p’, ’p53-p’), (’ATMa-p’, ’AKT-p’), (’ATMa-p’, ’KSRP-p’), (’ATMa-p’, ’CREB’),
    (’ATMa-p’, ’Chk2-p’), (’ATM-p’, ’MRN-p’), (’DSB’, ’MRN-p’), (’CREB’, ’ATM mRNA’),
    (’MRN-p’, ’ATMa-p’), (’CREB’, ’Wip mRNA’), (’p53-p’, ’Chk2 mRNA’), (’p53-p’, ’p21 mRNA’),
    (’p53-p’, ’PTEN mRNA’), (’p53-p’, ’Wip1 mRNA’), (’Wip1 mRNA’, ’Wip1’), (’KSRP-p’, ’pre-miR-16’),
    (’KSRP-p’, ’ATM-p’), (’Chk2 mRNA’, ’Chk2’), (’Chk2-p’, ’p53-p’), (’A20’, ’Bax mRNA’),
    (’Bax mRNA’, ’Bax’), (’Bax’, ’apoptosis’), (’p21 mRNA’, ’p21’), (’p21’, ’cell cycle arrest’),
    (’IR’, ’DSB’), (’PTEN’, ’PIP2’), (’PIP3’, ’AKT-p’), (’AKT-p’, ’Mdm2-p cyt’),
    (’IKKa’, ’NFkB’), (’NFkB’, ’IkBa mRNA’), (’NFkB’, ’p53 mRNA’), (’NFkB’, ’Wip mRNA’),
    (’IkBa mRNA’, ’IkBa’), (’IkBa’, ’Wip1 mRNA’), (’PTEN mRNA’, ’PTEN’), (’pre-miR-16’, ’miR-16’),
    (’A20 mRNA’, ’A20’), (’ATM mRNA’, ’ATM’), (’p53-p’, ’Bax mRNA’), (’p53-p’, ’Mdm2 mRNA’),
    (’Mdm2 mRNA’, ’Mdm2-p cyt’), (’Mdm2 cyt’, ’Mdm2-p cyt’), (’Mdm2-p cyt’, ’Mdm2-p nuc’),
    (’Chk2’, ’Chk2-p’), (’ATM-p’, ’ATM mRNA’), (’ATM’, ’ATM-p’), (’p53 mRNA’, ’p53’), (’p53’, ’p53-p’)
])

# Generate the adjacency matrix
adj_matrix = nx.adjacency_matrix(G).todense()
adj_matrix = np.array(adj_matrix)

# Generate node features using degree centrality, clustering coefficient, and betweenness centrality
degree_centrality = np.array([nx.degree_centrality(G)[node] for node in G.nodes()])
clustering_coefficient = np.array([nx.clustering(G.to_undirected())[node] for node in G.nodes()])
betweenness_centrality = np.array([nx.betweenness_centrality(G)[node] for node in G.nodes()])

# Combine the features into a feature matrix
feature_matrix = np.vstack((degree_centrality, clustering_coefficient, betweenness_centrality)).T

# Standardize features
scaler = StandardScaler()
feature_matrix = scaler.fit_transform(feature_matrix)

```

```
# Split the data into training (70%), validation (15%), and test (15%) sets
num_nodes = adj_matrix.shape[0]
indices = np.arange(num_nodes)
np.random.shuffle(indices)

train_idx = indices[:int(0.7 * num_nodes)]
val_idx = indices[int(0.7 * num_nodes):int(0.85 * num_nodes)]
test_idx = indices[int(0.85 * num_nodes):]

X_train = feature_matrix[train_idx].reshape(-1, 3, 1)
X_val = feature_matrix[val_idx].reshape(-1, 3, 1)
X_test = feature_matrix[test_idx].reshape(-1, 3, 1)

# Placeholder for target labels (binary classification: 0 or 1 for important nodes)
y_train = np.random.randint(0, 2, size=len(train_idx))
y_val = np.random.randint(0, 2, size=len(val_idx))
y_test = np.random.randint(0, 2, size=len(test_idx))

# Define the model
model = Sequential()

# Convolutional layer
model.add(Conv1D(filters=32, kernel_size=2, activation='relu', input_shape=(3, 1)))
model.add(MaxPooling1D(pool_size=2))

# LSTM layer
model.add(LSTM(128, return_sequences=False))

# Fully connected layer
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))

# Output layer
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Add early stopping to prevent overfitting
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

# Train the model
history = model.fit(X_train, y_train, epochs=30, batch_size=8, validation_data=(X_val, y_val), verbose=1, callbacks=[early_stopping])

# Evaluate the model on the test set
test_loss, test_accuracy = model.evaluate(X_test, y_test, verbose=0)
print(f"Test Accuracy: {test_accuracy:.2f}")

# Generate classification report
y_pred = (model.predict(X_test) > 0.5).astype(int)
print(classification_report(y_test, y_pred))
```

```

# Plot training history
plt.plot(history.history['accuracy'], label='train_accuracy')
plt.plot(history.history['val_accuracy'], label='val_accuracy')
plt.title('Model Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

# SHAP for model interpretability
explainer = shap.KernelExplainer(model.predict, X_train.reshape(X_train
    .shape[0], -1))
shap_values = explainer.shap_values(X_train.reshape(X_train.shape[0],
    -1))

# Plot SHAP summary
shap.summary_plot(shap_values, X_train.reshape(X_train.shape[0], -1),
    feature_names=['Degree', 'Clustering', 'Betweenness'])
```

Listing A.10: Improved Python code for model training and evaluation

Table A.1: Column Names for GDC TCGA-OV Clinical Data

Column Name	**Description**
Patient Identifier	Identifier to uniquely specify a patient.
Sample Identifier	A unique sample identifier.
Oncotree Code	Oncotree Code.
Cancer Type	Cancer Type.
Cancer Type Detailed	Cancer Type Detailed.
Tumor Type	Tumor Type.
Neoplasm Histologic Grade	Numeric value to express the degree of abnormality of cancer cells, a measure of differentiation and aggressiveness.
Tissue Prospective Collection Indicator	Tissue prospective collection indicator.
Tissue Retrospective Collection Indicator	Tissue retrospective collection indicator.
Tissue Source Site Code	Tissue Source Site Code.
Tumor Disease Anatomic Site	Text term that describes the anatomic site of the tumor or disease.
Aneuploidy Score	Aneuploidy Score.
Sample Type	The type of sample (e.g., normal, primary, metastasis, recurrence).
MSI MANTIS Score	MSI Score reported by MANTIS. Thresholds: MSI: >0.6, Indeterminate: 0.4-0.6, MSS: <0.4.
MSIsensor Score	MSI Score reported by MSIsensor. Thresholds: MSI: >10, Indeterminate: 4-10, MSS: <10.
Somatic Status	Somatic Status.
TMB (nonsynonymous)	TMB (nonsynonymous).
Tissue Source Site	Site collecting samples and clinical metadata for the Biospecimen Core Resource.
Mutation Status	Indicates whether mutations were detected in the sample (e.g., Yes/No).

A.1 Database Connection and Data Loading Script

The following Python script establishes a connection to a MySQL database and loads various genomic datasets into specified tables, as shown in Listing A.11.

```python
import mysql.connector
import pandas as pd

# Establish a connection to the MySQL database
db_config = {
    ’user’: ’root’,
    ’password’: ’root1’,
    ’host’: ’localhost’,
    ’database’: ’genomic_data_ovarian_cancer’
}

# Function to create a database connection
def create_connection(config):
    try:
        conn = mysql.connector.connect(**config)
        print("Connection to MySQL database was successful.")
        return conn
    except mysql.connector.Error as err:
        print(f"Error: {err}")
        return None

# Create a new database schema
def create_schema(cursor):
    cursor.execute("DROP TABLE IF EXISTS AnimalTF")
    cursor.execute("DROP TABLE IF EXISTS CellTalk")
    cursor.execute("DROP TABLE IF EXISTS Genomic_Data")
    cursor.execute("DROP TABLE IF EXISTS PathwayCommons")

    # Create AnimalTF Table
    cursor.execute("""
    CREATE TABLE AnimalTF (
        tf_id VARCHAR(20) PRIMARY KEY,
        gene_id VARCHAR(20),
        FOREIGN KEY (gene_id) REFERENCES Genomic_Data(gene_id)
    )
    """)

    # Create CellTalk Table
    cursor.execute("""
    CREATE TABLE CellTalk (
        interaction_id INT AUTO_INCREMENT PRIMARY KEY,
        source_gene_id VARCHAR(20),
        target_gene_id VARCHAR(20),
        FOREIGN KEY (source_gene_id) REFERENCES Genomic_Data(gene_id),
        FOREIGN KEY (target_gene_id) REFERENCES Genomic_Data(gene_id)
    )
    """)

    # Create Genomic_Data Table
    cursor.execute("""
    CREATE TABLE Genomic_Data (
        gene_id VARCHAR(20) PRIMARY KEY,
        gene_symbol VARCHAR(20),
        gene_name VARCHAR(100),
        annotation TEXT
    )
    """)

    # Create PathwayCommons Table
    cursor.execute("""
    CREATE TABLE PathwayCommons (
        pathway_id VARCHAR(20) PRIMARY KEY,
        pathway_name VARCHAR(100),
        gene_id VARCHAR(20),
        FOREIGN KEY (gene_id) REFERENCES Genomic_Data(gene_id)
    )
    """)

# Function to load data into the Genomic_Data table
def load_genomic_data(cursor, file_path):
    df = pd.read_csv(file_path)
    for index, row in df.iterrows():
        cursor.execute("""
        INSERT INTO Genomic_Data (gene_id, gene_symbol, gene_name, annotation)
        VALUES (%s, %s, %s, %s)
        """, (row[’gene_id’], row[’gene_symbol’], row[’gene_name’], row[’annotation’]))

# Function to load AnimalTF data into the table
def load_animaltf_data(cursor, file_path):
    df = pd.read_csv(file_path)
    for index, row in df.iterrows():
        cursor.execute("""
        INSERT INTO AnimalTF (tf_id, gene_id)
        VALUES (%s, %s)
        """, (row[’tf_id’], row[’gene_id’]))

# Function to load CellTalk data into the table
def load_celltalk_data(cursor, file_path):
    df = pd.read_csv(file_path)
    for index, row in df.iterrows():
        cursor.execute("""
        INSERT INTO CellTalk (source_gene_id, target_gene_id)
        VALUES (%s, %s)
        """, (row[’source_gene_id’], row[’target_gene_id’]))

# Function to load PathwayCommons data into the table
def load_pathwaycommons_data(cursor, file_path):
    df = pd.read_csv(file_path)
    for index, row in df.iterrows():
        cursor.execute("""
        INSERT INTO PathwayCommons (pathway_id, pathway_name, gene_id)
        VALUES (%s, %s, %s)
        """, (row[’pathway_id’], row[’pathway_name’], row[’gene_id’]))

# Main script execution
if __name__ == "__main__":
    connection = create_connection(db_config)

    if connection:
        cursor = connection.cursor()

        # Create database schema
        create_schema(cursor)

        # Load data into the database tables
        load_genomic_data(cursor, ’path_to_genomic_data.csv’)
        load_animaltf_data(cursor, ’path_to_animaltf_data.csv’)
        load_celltalk_data(cursor, ’path_to_celltalk_data.csv’)
        load_pathwaycommons_data(cursor, ’path_to_pathwaycommons_data.csv’)

        # Commit changes and close the connection
        connection.commit()
        cursor.close()
        connection.close()
        print("Database setup and data loading completed successfully.")
```

Listing A.11: MySQL Database Connection and Data Loading Script

```
# Import required libraries
import networkx as nx
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from collections import defaultdict

# Define mutation probabilities for each node
mutation_data = {
    ’p53’: 0.85, ’p53-p’: 0.82, ’Mdm2 mRNA’: 0.45,
    ’Mdm2 cyt’: 0.43, ’Mdm2-p cyt’: 0.41,
    # ... (remaining mutation data)
}

# Define interaction confidence scores
confidence_scores = {
    (’p53’, ’p53-p’): 0.9, (’p53 mRNA’, ’p53’): 0.8,
    # ... (remaining confidence scores)
}

# Create directed graph
G = nx.DiGraph()

# Initialize nodes with weight 0.5
for node in mutation_data:
    G.add_node(node, weight=0.5)

# Initialize edges with weight 1.0
edges = list(confidence_scores.keys())
G.add_edges_from([(u, v, {’weight’: 1.0}) for u, v in edges])

# Update edge weights based on mutation data and confidence scores
for u, v in G.edges():
    p_u = mutation_data.get(u, 0.1)  # Default 0.1 if missing
    p_v = mutation_data.get(v, 0.1)
    c_uv = confidence_scores.get((u, v), 0.5)  # Default 0.5 if missing
    G[u][v][’weight’] = p_u * p_v * c_uv  # Combined weight

# Normalize edge weights to [0,1] range
max_weight = max(data[’weight’] for _, _, data in G.edges(data=True))
for u, v in G.edges():
    G[u][v][’weight’] /= max_weight

def random_walk_with_restart(G, num_walks=5000, walk_length=15, restart_prob=0.15):
    """Perform random walks with restart probability"""
    nodes = list(G.nodes())
    node_visits = {node: 0 for node in nodes}

    for _ in range(num_walks):
        current_node = np.random.choice(nodes)
        node_visits[current_node] += 1

        for _ in range(walk_length):
            if np.random.rand() < restart_prob:
                current_node = np.random.choice(nodes)
            else:
                neighbors = list(G.successors(current_node))
                if neighbors:
                    weights = [G[current_node][n][’weight’] for n in neighbors]
                    probs = np.array(weights) / np.sum(weights)
                    current_node = np.random.choice(neighbors, p=probs)
                node_visits[current_node] += 1

    # Normalize visit counts
    total_visits = sum(node_visits.values())
    for node in node_visits:
        G.nodes[node][’weight’] = node_visits[node] / total_visits

    return G

# Execute the random walk algorithm
G = random_walk_with_restart(G)

# Generate and save results
node_weights = [(node, data[’weight’]) for node, data in G.nodes(data=True)]
node_weights.sort(key=lambda x: -x[1])  # Sort by significance

edge_weights = [(u, v, data[’weight’]) for u, v, data in G.edges(data=True)]
edge_weights.sort(key=lambda x: -x[2])  # Sort by interaction strength

# Save results to CSV files
pd.DataFrame(node_weights, columns=[’Node’, ’Weight’]).to_csv(’node_significance.csv’)
pd.DataFrame(edge_weights, columns=[’Source’, ’Target’, ’Weight’]).to_csv(’
    edge_strengths.csv’)

# Visualization code omitted for brevity
```

Listing A.12: Random walk algorithm implementation

```
import networkx as nx
import numpy as np
import matplotlib
matplotlib.use(’TkAgg’)  # Set the backend for interactive plotting
import matplotlib.pyplot as plt
from matplotlib import colors as mcolors
from networkx.algorithms import community

# Create a directed graph
G = nx.DiGraph()

# Define edges from the interaction data provided
edges = [
    (’p53’, ’p53-p’), (’p53 mRNA’, ’p53’), (’p53-p’, ’Mdm2 mRNA’),
    (’Mdm2 cyt’, ’Mdm2-p cyt’), (’Mdm2 mRNA’, ’Mdm2 cyt’),
    (’Mdm2-p cyt’, ’Mdm2-p nuc’), (’DSB’, ’ATM-p’),
    (’ATM mRNA’, ’ATM’), (’p53-p’, ’ATM mRNA’),
    (’ATMa-p’, ’p53-p’), (’ATMa-p’, ’AKT-p’),
    (’ATMa-p’, ’KSRP-p’), (’ATMa-p’, ’CREB’),
    (’ATMa-p’, ’Chk2-p’), (’ATM-p’, ’MRN-p’),
    (’DSB’, ’MRN-p’), (’CREB’, ’ATM mRNA’),
    (’MRN-p’, ’ATMa-p’), (’CREB’, ’Wip1 mRNA’),
    (’p53-p’, ’Chk2 mRNA’), (’p53-p’, ’Bax mRNA’),
    (’p53-p’, ’p21 mRNA’), (’p53-p’, ’PTEN mRNA’),
    (’p53-p’, ’Wip1 mRNA’), (’Wip1 mRNA’, ’Wip1’),
    (’pre-miR-16’, ’miR-16’), (’KSRP-p’, ’pre-miR-16’),
    (’Chk2 mRNA’, ’Chk2’), (’Chk2-p’, ’p53-p’),
    (’Bax mRNA’, ’Bax’), (’Bax’, ’apoptosis’),
    (’p21 mRNA’, ’p21’), (’p21’, ’cell cycle arrest’),
    (’IR’, ’DSB’), (’p53-p’, ’PTEN mRNA’),
    (’PTEN mRNA’, ’PTEN’), (’PTEN’, ’PIP2’),
    (’PIP2’, ’PIP3’), (’PIP3’, ’AKT-p’),
    (’AKT-p’, ’Mdm2-p cyt’), (’TNFa’, ’TNFR1’),
    (’TNFR1’, ’IKKKa’), (’IKKKa’, ’IKKa’),
    (’A20 mRNA’, ’A20 cyt’), (’IKKa’, ’NFkB’),
    (’NFkB’, ’IkBa mRNA’), (’NFkB’, ’A20 mRNA’),
    (’NFkB’, ’p53 mRNA’), (’IkBa mRNA’, ’IkBa’),
    (’NFkB’, ’Wip1 mRNA’)
]

G.add_edges_from(edges)

# Assign random weights to edges
```

```
for u, v in G.edges():
    G[u][v][’weight’] = np.random.rand()

# Random Walk Function
def random_walk(G, start_node, num_steps):
    current_node = start_node
    visiting_counts = {node: 0 for node in G.nodes()}

    for _ in range(num_steps):
        visiting_counts[current_node] += 1
        neighbors = list(G.neighbors(current_node))
        if not neighbors:
            break
        weights = [G[current_node][neighbor][’weight’] for neighbor in neighbors]
        current_node = np.random.choice(neighbors, p=np.array(weights) / sum(weights))

    return visiting_counts

# Parameters for the random walk
num_walks = 1000
num_steps = 50
start_node = ’p53’

# Perform random walks
total_visiting_counts = {node: 0 for node in G.nodes()}
convergence_data = []

for i in range(num_walks):
    counts = random_walk(G, start_node, num_steps)
    for node in total_visiting_counts:
        total_visiting_counts[node] += counts[node]

    # Record convergence data
    unique_nodes_visited = sum(1 for count in total_visiting_counts.values() if count >
    0)
    convergence_data.append(unique_nodes_visited / len(G.nodes()))

# Normalize visit counts
for node in total_visiting_counts:
    total_visiting_counts[node] /= num_walks

# Plot Convergence Rate
def plot_convergence_rate(convergence_data):
    plt.figure(figsize=(10, 5))
    plt.plot(convergence_data, marker=’o’)
    plt.title(’Convergence Rate of Random Walks’, fontsize=14)
    plt.xlabel(’Number of Simulations’, fontsize=12)
    plt.ylabel(’Proportion of Unique Nodes Visited’, fontsize=12)
    plt.grid()
    plt.tight_layout()
    plt.savefig(’convergence_rate.png’)
    plt.show()

# Function to plot the biological interaction network
def plot_biological_network(G, visiting_counts):
    fig, ax = plt.subplots(figsize=(14, 14))
    pos = nx.spring_layout(G, k=0.5, iterations=50)

    # Node sizes based on visiting counts
    node_sizes = [500 * visiting_counts[node] + 50 for node in G.nodes()]

    # Create a ScalarMappable for the color normalization
    norm = plt.Normalize(vmin=0, vmax=max(total_visiting_counts.values()))
    sm = plt.cm.ScalarMappable(cmap=’viridis’, norm=norm)
    sm.set_array([])

    # Draw nodes with improved aesthetics
    node_colors = [sm.to_rgba(visiting_counts[node]) for node in G.nodes()]
    nx.draw_networkx_nodes(G, pos, node_size=node_sizes, node_color=node_colors, alpha
    =0.9, edgecolors=’black’, ax=ax)

    # Draw edges with colors based on weights
    edge_colors = [plt.cm.Blues(G[u][v][’weight’]) for u, v in G.edges()]
    nx.draw_networkx_edges(G, pos, edge_color=edge_colors, alpha=0.8, style='solid', width=2, ax=ax)

    # Add labels for nodes
    labels = {node: node for node in G.nodes()}
    nx.draw_networkx_labels(G, pos, labels, font_size=10, font_color='black', ax=ax)

    # Add labels for edges with weights
    edge_labels = {(u, v): f"{G[u][v]['weight']:.2f}" for u, v in G.edges()}
    nx.draw_networkx_edge_labels(G, pos, edge_labels=edge_labels, font_color='red', ax=ax)

    # Add colorbar
    cbar = fig.colorbar(sm, ax=ax, label='Visit Count (normalized)', shrink=0.8, pad=0.02)

    plt.title('Enhanced Visualization of the Biological Interaction Network', fontsize=16)
    plt.axis('off')
    plt.savefig('biological_network.png')
    plt.show()

# Function to plot normalized visit counts
def plot_visit_counts(visiting_counts):
    plt.figure(figsize=(12, 6))
    nodes = list(visiting_counts.keys())
    counts = list(visiting_counts.values())

    plt.bar(nodes, counts, color='royalblue')
    plt.xlabel('Nodes', fontsize=12)
    plt.ylabel('Normalized Visit Count', fontsize=12)
    plt.title('Normalized Visit Counts per Node', fontsize=14)
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.savefig('normalized_visit_counts.png')
    plt.show()

# Function to plot edge weights distribution
def plot_edge_weights_distribution(G):
    edge_weights = [G[u][v]['weight'] for u, v in G.edges()]

    plt.figure(figsize=(12, 6))
    plt.hist(edge_weights, bins=20, color='lightcoral', edgecolor='black')
    plt.xlabel('Edge Weight', fontsize=12)
    plt.ylabel('Frequency', fontsize=12)
    plt.title('Distribution of Edge Weights', fontsize=14)
    plt.tight_layout()
    plt.savefig('edge_weights_distribution.png')
    plt.show()

# Function to plot community sizes
def plot_community_sizes(G):
    communities = community.greedy_modularity_communities(G)
    community_sizes = [len(comm) for comm in communities]

    plt.figure(figsize=(12, 6))
    plt.bar(range(len(community_sizes)), community_sizes, color='mediumseagreen')
    plt.xlabel('Community Index', fontsize=12)
    plt.ylabel('Community Size', fontsize=12)
    plt.title('Sizes of Detected Communities', fontsize=14)
    plt.tight_layout()
    plt.savefig('community_sizes.png')
    plt.show()

# Run all plots
plot_convergence_rate(convergence_data)
plot_biological_network(G, total_visiting_counts)
plot_visit_counts(total_visiting_counts)
plot_edge_weights_distribution(G)
plot_community_sizes(G)
```

Listing A.13: Random Walk Analysis in a Directed Graph

Supplementary Code

The following Python code implements a Boolean network simulation, including the creation of nodes and edges, state initialization, and updates through iterations. It also provides visualizations of the node states over time, identifies pivotal nodes, and plots the network dynamics.

```python
import matplotlib.pyplot as plt
import random as rd
import networkx as nx
import numpy as np

# Define Graph
G = nx.DiGraph()

# Add Nodes (Part 1)
nodes_data = [
    ('TNFa', {'property': "trans", 'weight': rd.randint(0, 1)}),
    ('TNFR1', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('IKKKa', {'property': "pro", 'weight': rd.randint(0, 1)}),
    ('IKKa', {'property': "pro", 'weight': rd.randint(0, 1)}),
    ('NFkB (TF)', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('p53 (TF)', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('p53-p', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('A20 mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('A20', {'property': "pro", 'weight': rd.randint(0, 1)}),
    ('IkBa mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('Mdm2 cyt', {'property': 'pro', 'weight': rd.randint(0, 1)}),
    ('IkBa', {'property': "pro", 'weight': rd.randint(0, 1)}),
    ('Wip mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('Wip1', {'property': 'pro', 'weight': rd.randint(0, 1)}),
    ('Wip1 mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('p53 mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('ATM mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('ATM', {'property': 'gen', 'weight': rd.randint(0, 1)}),
    ('ATM-p', {'property': 'gen', 'weight': rd.randint(0, 1)}),
    ('ATMa-p', {'property': 'gen', 'weight': rd.randint(0, 1)}),
    ('MRN-p', {'property': 'pro', 'weight': rd.randint(0, 1)}),
    ('Chk2-p', {'property': 'pro', 'weight': rd.randint(0, 1)}),
    ('CREB (TF)', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('KSRP-p', {'property': 'prot', 'weight': rd.randint(0, 1)}),
    ('AKT-p', {'property': 'prot', 'weight': rd.randint(0, 1)}),
    ('Mdm2-p cyt', {'property': 'prot', 'weight': rd.randint(0, 1)}),
    ('Mdm2-p nuc', {'property': 'prot', 'weight': rd.randint(0, 1)}),
    ('PTEN mRNA (Genomic)', {'property': 'gen', 'weight': rd.randint(0,
    1)}),
    ('PTEN', {'property': 'pro', 'weight': rd.randint(0, 1)}),
    ('PIP2', {'property': "lipo", 'weight': rd.randint(0, 1)}),
    ('PIP3', {'property': "lipo", 'weight': rd.randint(0, 1)}),
    ('Bax mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('Bax', {'property': 'pro', 'weight': rd.randint(0, 1)}),
    ('apoptosis', {'property': "gen", 'weight': rd.randint(0, 1)}),
    ('Mdm2 mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('p21 mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('p21', {'property': 'prot', 'weight': rd.randint(0, 1)}),
    ('cell cycle arrest', {'property': "trans", 'weight': rd.randint(0,
    1)}),
    ('Chk2 mRNA', {'property': 'trans', 'weight': rd.randint(0, 1)}),
    ('Chk2', {'property': 'pro', 'weight': rd.randint(0, 1)}),
    (’IR’, {’property’: ’gen’, ’weight’: rd.randint(0, 1)}),
    (’DSB’, {’property’: "gen", ’weight’: rd.randint(0, 1)}),
    (’miR-16’, {’property’: ’trans’, ’weight’: rd.randint(0, 1)}),
    (’pre-miR-16’, {’property’: ’trans’, ’weight’: rd.randint(0, 1)}),
]

G.add_nodes_from(nodes_data)

# Initialize ’weight’ attribute for all nodes
for node, data in nodes_data:
    G.nodes[node][’weight’] = data[’weight’]

# Add Edges (Part 1 and 2)
edges_data = [
    (’TNFa’, ’TNFR1’, {’weight’: rd.randint(0, 1)}),
    (’TNFR1’, ’IKKKa’, {’weight’: rd.randint(0, 1)}),
    (’IKKKa’, ’IKKa’, {’weight’: rd.randint(0, 1)}),
    (’IKKa’, ’NFkB (TF)’, {’weight’: rd.randint(0, 1)}),
    (’TNFR1’, ’NFkB (TF)’, {’weight’: rd.randint(0, 1)}),
    (’ATMa-p’, ’IKKa’, {’weight’: rd.randint(0, 1)}),
    (’p53 (TF)’, ’p53-p’, {’weight’: rd.randint(0, 1)}),
    (’PIP2’, ’PIP3’, {’weight’: rd.randint(0, 1)}),
    (’p53 mRNA’, ’p53’, {’weight’: rd.randint(0, 1)}),
    (’DSB’, ’ATM-p’, {’weight’: rd.randint(0, 1)}),
    (’p53-p’, ’ATM mRNA’, {’weight’: rd.randint(0, 1)}),
    (’ATMa-p’, ’p53-p’, {’weight’: rd.randint(0, 1)}),
    (’ATMa-p’, ’AKT-p’, {’weight’: rd.randint(0, 1)}),
    (’ATMa-p’, ’KSRP-p’, {’weight’: rd.randint(0, 1)}),
    (’ATMa-p’, ’CREB (TF)’, {’weight’: rd.randint(0, 1)}),
    (’ATMa-p’, ’Chk2-p’, {’weight’: rd.randint(0, 1)}),
    (’ATM-p’, ’MRN-p’, {’weight’: rd.randint(0, 1)}),
    (’DSB’, ’MRN-p’, {’weight’: rd.randint(0, 1)}),
    (’CREB (TF)’, ’ATM mRNA’, {’weight’: rd.randint(0, 1)}),
    (’MRN-p’, ’ATMa-p’, {’weight’: rd.randint(0, 1)}),
    (’CREB (TF)’, ’Wip mRNA’, {’weight’: rd.randint(0, 1)}),
    (’p53-p’, ’Chk2 mRNA’, {’weight’: rd.randint(0, 1)}),
    (’p53-p’, ’p21 mRNA’, {’weight’: rd.randint(0, 1)}),
    (’p53-p’, ’PTEN mRNA (Genomic)’, {’weight’: rd.randint(0, 1)}),
    (’p53-p’, ’Wip1 mRNA’, {’weight’: rd.randint(0, 1)}),
    (’Wip1 mRNA’, ’Wip1’, {’weight’: rd.randint(0, 1)}),
    (’KSRP-p’, ’pre-miR-16’, {’weight’: rd.randint(0, 1)}),
    (’ATM-p’, ’Chk2’, {’weight’: rd.randint(0, 1)}),
    (’Chk2-p’, ’Mdm2 mRNA’, {’weight’: rd.randint(0, 1)}),
    (’Mdm2 mRNA’, ’Mdm2 cyt’, {’weight’: rd.randint(0, 1)}),
    (’Mdm2 cyt’, ’Mdm2-p nuc’, {’weight’: rd.randint(0, 1)}),
    (’Mdm2 cyt’, ’Bax’, {’weight’: rd.randint(0, 1)}),
    (’Mdm2 cyt’, ’Mdm2-p cyt’, {’weight’: rd.randint(0, 1)}),
    (’Mdm2 cyt’, ’p21’, {’weight’: rd.randint(0, 1)}),
    (’p21’, ’cell cycle arrest’, {’weight’: rd.randint(0, 1)}),
    (’p21’, ’apoptosis’, {’weight’: rd.randint(0, 1)}),
    (’Bax’, ’apoptosis’, {’weight’: rd.randint(0, 1)}),
    (’Mdm2-p nuc’, ’Mdm2-p cyt’, {’weight’: rd.randint(0, 1)}),
    (’PTEN’, ’PIP2’, {’weight’: rd.randint(0, 1)}),
]

G.add_edges_from(edges_data)

# Initialize edge weights
```

```
for u, v, data in G.edges(data=True):
    data[’weight’] = rd.randint(0, 1)

# Run simulation
time_steps = 10
states = {}
node_states = {node: rd.choice([0, 1]) for node in G.nodes}

for t in range(time_steps):
    states[t] = node_states.copy()

    for node in G.nodes:
        inputs = list(G.predecessors(node))
        total_input = sum(node_states[input_node] * G[input_node][node][’weight’] for input_node in inputs)
        if total_input > 0:
            node_states[node] = 1
        else:
            node_states[node] = 0

# Visualization
for t in range(time_steps):
    plt.bar(states[t].keys(), states[t].values(), label=f’Time {t}’)
    plt.ylim(-0.5, 1.5)
    plt.title(’Node States Over Time’)
    plt.xlabel(’Nodes’)
    plt.ylabel(’State’)
    plt.xticks(rotation=45)
    plt.legend()
    plt.tight_layout()
    plt.show()

# Identify pivotal nodes
def identify_pivotal_nodes(G):
    pivotal_nodes = []
    for node in G.nodes:
        total_input = sum(G.predecessors(node), key=lambda x: G[x][node][’weight’])
        if total_input > 0:
            pivotal_nodes.append(node)
    return pivotal_nodes

pivotal_nodes = identify_pivotal_nodes(G)
print("Pivotal nodes:", pivotal_nodes)

# Plot network dynamics
pos = nx.spring_layout(G)
nx.draw(G, pos, with_labels=True, node_color=’lightblue’, node_size=1000)
plt.title("Network Dynamics")
plt.show()
```

Table A.2: Overview of Combined Data: Sample Rows from Each Downloaded CSV File

GDC Downloaded CSV File Data						
Gene ID	**Gene Name**	**Gene Type**	**Unstranded**	**Stranded First**	**Stranded Second**	**TPM Unstrande**
N_unmapped			11394772	11394772	11394772	
N_multimapping			14179883	14179883	14179883	
N_noFeature			6530100	42884641	43025237	
N_ambiguous			8298602	2284369	2280443	
ENSG00000000003.15	TSPAN6	protein_coding	4768	2441	2327	38.2031
ENSG00000000005.6	TNMD	protein_coding	4	0	4	0.0985
ENSG00000000419.13	DPM1	protein_coding	2397	1170	1227	72.1765
ENSG00000000457.14	SCYL3	protein_coding	1621	1355	1441	8.5593
ENSG00000000460.17	C1orf112	protein_coding	329	802	795	2.0029
CellTalk Database Data						
LR Pair	**Ligand Gene Symbol**	**Receptor Gene Symbol**	**Ligand Gene ID**	**Receptor Gene ID**	**Ligand Ensembl Protein ID**	**Receptor Ense Protein ID**
SEMA3F_PLXNA3	SEMA3F	PLXNA3	6405	55558	ENSP00000002829	ENSP00000358696
SEMA3F_PLXNA1	SEMA3F	PLXNA1	6405	5361	ENSP00000002829	ENSP00000377061
SEMA3F_NRP1	SEMA3F	NRP1	6405	8829	ENSP00000002829	ENSP00000265371
SEMA3F_NRP2	SEMA3F	NRP2	6405	8828	ENSP00000002829	ENSP00000353582
CX3CL1_CX3CR1	CX3CL1	CX3CR1	6376	1524	ENSP00000006053	ENSP00000351059
AnimalTF Database Data						
Species	**Symbol**	**Ensembl**	**Family**	**Protein**	**Entrez ID**	
Homo_sapiens	ZBTB8B	ENSG00000273274	ZBTB	ENSP00000476499	728116	
Homo_sapiens	GSX2	ENSG00000180613	Homeobox	ENSP00000319118;ENSP00000433522	170825	
Homo_sapiens	TBX2	ENSG00000121068	T-box	ENSP00000404781;ENSP00000240328	6909	
Homo_sapiens	PAX8	ENSG00000125618	PAX	ENSP00000395498;ENSP00000263335;ENSP00000380768;ENSP0000026	7849	
Homo_sapiens	CREB3L1	ENSG00000157613	TF_bZIP	ENSP00000481956;ENSP00000436574	90993	
Homo_sapiens	NKX6-1	ENSG00000163623	Homeobox	ENSP00000295886;ENSP00000449761	4825	
PathwayCommons Downloaded CSV File Data						
Participant A	**Interaction Type**	**Participant B**	**Interaction Data Source**	**Interaction PubMed ID**	**Pathway Names**	

Table A.2: (Continued)

Gene ID	**Gene Name**	**Gene Type**	**Unstranded**	**Stranded First**	**Stranded Second**	**TPM Unstrande**
A4GALT	catalysis-precedes	ABO	KEGG		Glycosphingolipid biosynthesis - globo series; Glycosphingolipid biosynthesis - lacto and neolacto series; Metabolic pathways	
A4GALT	catalysis-precedes	AK3	KEGG		Glycosphingolipid biosynthesis - globo series; Metabolic pathways; Pyrimidine metabolism	
A4GALT	catalysis-precedes	ALG13	KEGG		Glycosphingolipid biosynthesis - globo series; Glycosphingolipid biosynthesis - lacto and neolacto series; Metabolic pathways	

Table A.3: Dataset of Biochemical Interactions

Source Node	**Target Node**	**Interaction Type**
IR	DSB	interacts_with
DSB	DNA	interacts_with
DNA	DSB	interacts_with
DNA	p53mRNA	interacts_with
p53mRNA	p53	interacts_with
p53	p53-p	interacts_with
p53-p	p53	interacts_with
DNA	PTENmRNA nuc	interacts_with
DNA	Mdm2mRNA nuc	interacts_with
DNA	PTENmRNA nuc	interacts_with
PTENmRNA nuc	PTEN cyt	interacts_with
PTEN cyt	PIP2	interacts_with
PIP3	PIP2	interacts_with
PIP2	PIP3	interacts_with
Mdm2mRNA nuc	Mdm2 cyt	interacts_with
Mdm2-p cyt	Mdm2 cyt	interacts_with
Mdm2 cyt	Mdm2-p cyt	interacts_with
AKT-p cyt	AKT cyt	interacts_with
PIP3	AKT-p cyt	interacts_with
AKT-p cyt	Mdm2-p cyt	interacts_with
AKT cyt	AKT-p cyt	interacts_with
Mdm2 cyt	Mdm2-p cyt	interacts_with
Mdm2 cyt	Mdm2-p-p nuc	interacts_with
Mdm2-p cyt	Mdm2-p nuc	interacts_with
p53-p	Bax mRNA	interacts_with
p53-p	p21 mRNA	interacts_with
p53-p	PTENmRNA nuc	interacts_with
p53-p	IkBα	interacts_with
p53-p	A20 mRNA	interacts_with
p53-p	Mdm2mRNA nuc	interacts_with
Mdm2-p nuc	p53	interacts_with
Mdm2-p-p nuc	Mdm2-p nuc	interacts_with
Mdm2-p nuc	Mdm2-p-p nuc	interacts_with

Continued on next page

Table A.3 – continued from previous page

Source Node	Target Node	Interaction Type
Mdm2-p nuc	p53	interacts_with
p53	DSB	interacts_with
p53	DNA	interacts_with
DNA	Bax mRNA nuc	interacts_with
DNA	p21 mRNA nuc	interacts_with
p21 mRNA nuc	Bax mRNA nuc	interacts_with
p21	p21 mRNA nuc	interacts_with
p21 mRNA nuc	cell cycle arrest	interacts_with
Bax	p21	interacts_with
Bax mRNA nuc	Bax	interacts_with
Bax	apoptosis	interacts_with
DNA	A20 mRNA nuc	interacts_with
DNA	IkBα mRNA nuc	interacts_with
IkBα mRNA nuc	IkBα nuc	interacts_with
A20 mRNA nuc	A20 cyt	interacts_with
A20 cyt	IKKKa cyt	interacts_with
A20 cyt	IKKi cyt	interacts_with
A20 cyt	IKKa cyt	interacts_with
IKKi cyt	IKKii cyt	interacts_with
A20 cyt	IKKKa cyt	interacts_with
IKKKa cyt	IKKKn cyt	interacts_with
IKKKn cyt	IKKKa cyt	interacts_with
IKKKa cyt	IKKa cyt	interacts_with
IKKa cyt	IKKi cyt	interacts_with
IKKii cyt	IKKKn cyt	interacts_with
IKKn cyt	IKKa cyt	interacts_with
IKKa cyt	IkBα:NFkB cyt	interacts_with
IkBα:NFkB cyt	NFkB cyt	interacts_with
IkBα:NFkB cyt	IkBα * cyt	interacts_with
NFkB cyt	IkBα:NFkB cyt	interacts_with
IkBα cyt	IkBα:NFkB cyt	interacts_with
IkBα cyt	IkBα nuc	interacts_with
NFkB cyt	NFkB nuc	interacts_with
NFkB nuc	IkBα mRNA nuc	interacts_with

Continued on next page

Table A.3 – continued from previous page

Source Node	**Target Node**	**Interaction Type**
NFkB nuc	IkBα:NFkB nuc	interacts_with
IkBα nuc	IkBα:NFkB nuc	interacts_with
IkBα:NFkB nuc	IkBα:NFkB cyt	interacts_with
IkBα mRNA	IkBα	interacts_with
NFkB nuc	A20 mRNA nuc	interacts_with
NFkB nuc	p53mRNA nuc	interacts_with
DNA	pre-mRNA-16 nuc	interacts_with
DNA	Wip1 mRNA nuc	interacts_with
pre-mRNA-16 nuc	miR-16 nuc	interacts_with
miR-16 nuc	Wip1 mRNA nuc	interacts_with
Wip1 mRNA nuc	Wip1 nuc	interacts_with
Wip1 nuc	p53	interacts_with
Wip1 nuc	Wip1 mRNA	interacts_with
Wip1 nuc	A20 mRNA	interacts_with
Wip1 nuc	IkBα mRNA	interacts_with
Wip1 nuc	Mdm2-p nuc	interacts_with
Wip1 nuc	ChK2 nuc	interacts_with
Wip1 mRNA nuc	ChK2 mRNA	interacts_with
Wip1 nuc	ATM nuc	interacts_with
Wip1 nuc	ATM-p nuc	interacts_with
KSRP cyt	KSRP-p cyt	interacts_with
KSRP-p cyt	KSRP cyt	interacts_with
KSRP-p cyt	KSRP-p nuc	interacts_with
KSRP-p nuc	pre-mRNA-16 nuc	interacts_with
DNA	ChK2 mRNA nuc	interacts_with
ChK2 mRNA nuc	ChK2 nuc	interacts_with
ChK2 nuc	ChK2-p nuc	interacts_with
ChK2-p nuc	Chk2 nuc	interacts_with
ChK2-p nuc	p53-p	interacts_with
ChK2-p nuc	Mdm2-p-p nuc	interacts_with
ChK2-p nuc	Mdm2-p nuc	interacts_with
ChK2-p nuc	Mdm2 cyt	interacts_with
ChK2-p nuc	Mdm2-p cyt	interacts_with
DNA	ATM mRNA nuc	interacts_with

Continued on next page

Table A.3 – continued from previous page

Source Node	**Target Node**	**Interaction Type**
ATM mRNA nuc	ATM nuc	interacts_with
ATM-p nuc	ATM nuc	interacts_with
ATM nuc	ATM-p nuc	interacts_with
ATMa-p nuc	ATM-p nuc	interacts_with
ATM-p nuc	ATMa-p nuc	interacts_with
ATMa-p nuc	p53-p	interacts_with
ATMa-p nuc	IKKa cyt	interacts_with
ATMa-p nuc	Mdm2-p-p nuc	interacts_with
ATMa-p nuc	KSRP-p cyt	interacts_with
ATMa-p nuc	AKT-p cyt	interacts_with
ATMa-p nuc	CREB nuc	interacts_with
CREB nuc	CREB-p nuc	interacts_with
CREB-p nuc	CREB nuc	interacts_with
CREB nuc	Wip1 mRNA nuc	interacts_with
CREB nuc	ATM mRNA nuc	interacts_with
ATM-p nuc	MRN-p nuc	interacts_with
DSB nuc	MRN-p nuc	interacts_with
MRN-p nuc	MRN nuc	interacts_with
MRN nuc	MRN-p nuc	interacts_with
MRN-p nuc	ATMa-p nuc	interacts_with
$TNF\alpha$	TNFR1 cyt	interacts_with
TNFR1 cyt	IKKKa cyt	interacts_with

Source Node	Target Node	Interaction Type
IR	DSB	PPI: interacts_with
DSB	DNA	PPI: interacts_with
DNA	DSB	PPI: interacts_with
DNA	p53mRNA	TF-regulates: interacts_with
p53mRNA	p53	TF-regulates: interacts_with
p53	p53-p	TF-regulates: interacts_with
p53-p	p53	TF-regulates: interacts_with
DNA	PTENmRNA_nuc	TF-regulates: interacts_with
DNA	Mdm2mRNA_nuc	TF-regulates: interacts_with
DNA	PTENmRNA_nuc	TF-regulates: interacts_with
PTENmRNA_nuc	PTEN_cyt	PPI: interacts_with
PTEN_cyt	PIP2	PPI: interacts_with
PIP3	PIP2	PPI: interacts_with
PIP2	PIP3	PPI: interacts_with
Mdm2mRNA_nuc	Mdm2_cyt	PPI: interacts_with
Mdm2-p_cyt	Mdm2_cyt	PPI: interacts_with
Mdm2_cyt	Mdm2-p_cyt	PPI: interacts_with
AKT-p_cyt	AKT_cyt	PPI: interacts_with
PIP3	AKT-p_cyt	PPI: interacts_with
AKT-p_cyt	Mdm2-p_cyt	PPI: interacts_with
AKT_cyt	AKT-p_cyt	PPI: interacts_with
Mdm2_cyt	Mdm2-p_cyt	PPI: interacts_with
Mdm2_cyt	Mdm2-p-p_nuc	PPI: interacts_with
Mdm2-p_cyt	Mdm2-p_nuc	PPI: interacts_with
p53-p	Bax_mRNA	TF-regulates: interacts_with
p53-p	p21_mRNA	TF-regulates: interacts_with
p53-p	PTENmRNA_nuc	TF-regulates: interacts_with
p53-p	IkB	TF-regulates: interacts_with
p53-p	A20_mRNA	TF-regulates: interacts_with
p53-p	Mdm2mRNA_nuc	TF-regulates: interacts_with
Mdm2-p_nuc	p53	TF-regulates: interacts_with
Mdm2-p-p_nuc	Mdm2-p_nuc	PPI: interacts_with
Mdm2-p_nuc	Mdm2-p-p_nuc	PPI: interacts_with
Mdm2-p_nuc	p53	TF-regulates: interacts_with
p53	DSB	PPI: interacts_with
p53	DNA	PPI: interacts_with
DNA	Bax_mRNA_nuc	TF-regulates: interacts_with
DNA	p21_mRNA_nuc	TF-regulates: interacts_with
p21_mRNA_nuc	Bax_mRNA_nuc	TF-regulates: interacts_with
p21	p21_mRNA_nuc	TF-regulates: interacts_with
p21_mRNA_nuc	cell_cycle_arrest	TF-regulates: interacts_with
Bax	p21	TF-regulates: interacts_with
Bax_mRNA_nuc	Bax	PPI: interacts_with
Bax	apoptosis	PPI: interacts_with
DNA	A20_mRNA_nuc	TF-regulates: interacts_with
DNA	IkB_mRNA_nuc	TF-regulates: interacts_with

Define Python style for syntax highlighting

A.2 Supplementary Materials

A.2.1 Network Construction and Analysis Code

```python
import networkx as nx
import matplotlib.pyplot as plt
import random

# ==============================================
# CELL CYCLE NETWORK CONSTRUCTION
# ==============================================
G_cell_cycle = nx.DiGraph()

# Core Cell Cycle Nodes (79 nodes)
cell_cycle_nodes = [
    # Cyclins & CDKs
    ’Cyclin D1’, ’Cyclin D2’, ’Cyclin D3’, ’Cyclin E1’, ’Cyclin E2’,
    ’Cyclin A1’, ’Cyclin A2’, ’Cyclin B1’, ’Cyclin B2’, ’Cyclin B3’,
    ’CDK1’, ’CDK2’, ’CDK4’, ’CDK6’,
    # CDK Inhibitors
    ’p21’, ’p27’, ’p57’, ’p16’, ’p15’, ’p18’, ’p19’,
    # Rb-E2F Pathway
    ’Rb’, ’Rb-p’, ’E2F1’, ’E2F2’, ’E2F3’, ’E2F4’, ’E2F5’, ’DP1’, ’DP2’,
    # DNA Damage Response
    ’ATM’, ’ATR’, ’Chk1’, ’Chk2’, ’p53’, ’Mdm2’, ’Wip1’, ’14-3-3’,
    ’CDC25A’, ’CDC25B’, ’CDC25C’,
    # Checkpoints (15 nodes)
    ’BubR1’, ’Bub1’, ’Bub3’, ’Mad1’, ’Mad2’, ’Aurora A’, ’Aurora B’,
    ’PLK1’, ’WEE1’, ’MYT1’,
    # Ubiquitin Ligases (7 nodes)
    ’APC/C’, ’CDC20’, ’CDH1’, ’SCF’, ’SKP2’, ’CUL1’, ’FBXW7’,
    # Transcription Factors (7 nodes)
    ’MYC’, ’FOXM1’, ’HIF1A’, ’NF-kB’, ’STAT3’, ’YAP1’, ’NOTCH1’,
    # Phenotypes (7 nodes)
    ’G1_phase’, ’S_phase’, ’G2_phase’, ’M_phase’,
    ’cell_cycle_arrest’, ’senescence’, ’apoptosis’
]

G_cell_cycle.add_nodes_from(cell_cycle_nodes)

# Expanded Interactions (182 edges)
cell_cycle_edges = [
    # Cyclin-CDK Complexes
    (’Cyclin D1’, ’CDK4’, ’binding’), (’Cyclin D1’, ’CDK6’, ’binding’),
    (’Cyclin E1’, ’CDK2’, ’binding’), (’Cyclin A1’, ’CDK2’, ’binding’),
    (’Cyclin B1’, ’CDK1’, ’binding’),

    # Rb-E2F Regulation
    (’Cyclin D1/CDK4’, ’Rb’, ’phosphorylation’),
    (’Cyclin E1/CDK2’, ’Rb’, ’phosphorylation’),
    (’Rb’, ’E2F1’, ’inhibition’), (’Rb-p’, ’E2F1’, ’release’),
    (’E2F1’, ’Cyclin E1’, ’transcription’),
    (’E2F1’, ’Cyclin A1’, ’transcription’),

    # CDK Inhibitors
    (’p21’, ’Cyclin E1/CDK2’, ’inhibition’),
    (’p21’, ’Cyclin A1/CDK2’, ’inhibition’),
    (’p27’, ’Cyclin D1/CDK4’, ’inhibition’),
    (’p16’, ’CDK4’, ’inhibition’),
    (’p53’, ’p21’, ’transcription’),

    # DNA Damage Response
    (’ATM’, ’Chk2’, ’phosphorylation’),
    (’ATR’, ’Chk1’, ’phosphorylation’),
    (’Chk1’, ’CDC25A’, ’inhibition’),
    (’Chk2’, ’CDC25C’, ’inhibition’),
    (’Chk1’, ’p53’, ’phosphorylation’),
    (’p53’, ’Mdm2’, ’transcription’),
    (’Mdm2’, ’p53’, ’ubiquitination’),
    (’Wip1’, ’ATM’, ’dephosphorylation’),

    # Checkpoints
    (’BubR1’, ’APC/C’, ’inhibition’),
    (’Mad2’, ’CDC20’, ’inhibition’),
    (’Aurora B’, ’BubR1’, ’phosphorylation’),
    (’PLK1’, ’CDC25C’, ’activation’),

    # Ubiquitin-Mediated Degradation
    (’APC/C-CDC20’, ’Cyclin B1’, ’degradation’),
    (’APC/C-CDH1’, ’Cyclin A1’, ’degradation’),
    (’SCF-SKP2’, ’p27’, ’degradation’),
    (’FBXW7’, ’Cyclin E1’, ’degradation’),

    # Cross-Talk with Other Pathways
    (’MYC’, ’Cyclin D1’, ’transcription’),
    (’NF-kB’, ’Cyclin D1’, ’transcription’),
    (’STAT3’, ’Cyclin D1’, ’transcription’),
    (’YAP1’, ’Cyclin E1’, ’transcription’),
    (’NOTCH1’, ’Cyclin D1’, ’transcription’),
    (’HIF1A’, ’Cyclin D1’, ’transcription’),

    # Phenotypic Outcomes
    (’Cyclin B1/CDK1’, ’M_phase’, ’initiation’),
    (’Cyclin D1/CDK4’, ’G1_phase’, ’progression’),
    (’p21’, ’cell_cycle_arrest’, ’induction’),
    (’p53’, ’apoptosis’, ’induction’),
]

# Add edges with random weights
for source, target, interaction in cell_cycle_edges:
    G_cell_cycle.add_edge(source, target,
                         weight=random.uniform(0.5, 2.0),
                         interaction=interaction)

# Add 150 random edges to increase complexity
for _ in range(150):
    src, tgt = random.sample(cell_cycle_nodes, 2)
    if not G_cell_cycle.has_edge(src, tgt):
        G_cell_cycle.add_edge(src, tgt,
                             weight=random.uniform(0.1, 0.5),
                             interaction=’random_crosstalk’)

# ============================================
# MAPK NETWORK CONSTRUCTION
# ============================================
G_mapk = nx.DiGraph()

# Core MAPK Nodes (71 nodes)
mapk_nodes = [
    # Growth Factor Receptors (10 nodes)
    ’EGFR’, ’HER2’, ’HER3’, ’HER4’, ’IGF1R’, ’INSR’,
    ’PDGFR’, ’FGFR’, ’MET’, ’NTRK’,
    # RAS-RAF-MEK-ERK Pathway (12 nodes)
    ’GRB2’, ’SOS1’, ’RAS’, ’HRAS’, ’KRAS’, ’NRAS’,
    ’RAF1’, ’BRAF’, ’MEK1’, ’MEK2’, ’ERK1’, ’ERK2’,
    # PI3K-AKT-mTOR Pathway (13 nodes)
    ’PI3K’, ’PIK3CA’, ’PIK3CB’, ’PIK3CD’, ’PTEN’,
    ’PIP2’, ’PIP3’, ’AKT1’, ’AKT2’, ’AKT3’,
    ’PDK1’, ’mTORC1’, ’mTORC2’,
    # Downstream Effectors (12 nodes)
    ’RSK1’, ’RSK2’, ’RSK3’, ’MSK1’, ’MSK2’,
    ’MNK1’, ’MNK2’, ’ELK1’, ’CREB’, ’c-FOS’, ’c-JUN’, ’MYC’,
    # Negative Regulators (7 nodes)
    ’DUSP1’, ’DUSP6’, ’SPRY2’, ’SPRY4’, ’PP2A’, ’PTPN11’, ’NF1’,
    # Cross-Talk Nodes (10 nodes)
    ’GSK3B’, ’FOXO1’, ’FOXO3’, ’BAD’, ’BCL2’,
    ’BCLXL’, ’p70S6K’, ’4E-BP1’, ’eIF4E’, ’HIF1A’,
    # Phenotypes (5 nodes)
    ’cell_proliferation’, ’cell_survival’,
    ’migration’, ’invasion’, ’angiogenesis’
]
```

```

G_mapk.add_nodes_from(mapk_nodes)

# Expanded Interactions (175 edges)
mapk_edges = [
    # EGFR-RAS-ERK Pathway
    ('EGFR', 'GRB2', 'binding'), ('GRB2', 'SOS1', 'recruitment'),
    ('SOS1', 'RAS', 'activation'), ('RAS', 'RAF1', 'activation'),
    ('RAF1', 'MEK1', 'phosphorylation'),
    ('MEK1', 'ERK1', 'phosphorylation'),
    ('ERK1', 'RSK1', 'activation'),
    ('ERK1', 'c-FOS', 'phosphorylation'),
    ('ERK1', 'MYC', 'stabilization'),

    # PI3K-AKT Pathway
    ('EGFR', 'PI3K', 'activation'),
    ('PI3K', 'PIP3', 'production'),
    ('PIP3', 'AKT1', 'recruitment'),
    ('PDK1', 'AKT1', 'phosphorylation'),
    ('AKT1', 'mTORC1', 'activation'),
    ('mTORC1', 'p70S6K', 'activation'),
    ('mTORC1', '4E-BP1', 'inhibition'),
    ('AKT1', 'GSK3B', 'inhibition'),
    ('AKT1', 'FOXO1', 'inhibition'),

    # Negative Feedback
    ('DUSP1', 'ERK1', 'dephosphorylation'),
    ('SPRY2', 'GRB2', 'inhibition'),
    ('PP2A', 'AKT1', 'dephosphorylation'),
    ('PTEN', 'PIP3', 'dephosphorylation'),

    # Cross-Talk with Other Pathways
    ('ERK1', 'AR', 'phosphorylation'),
    ('AKT1', 'AR', 'phosphorylation'),
    ('ERK1', 'ESR1', 'phosphorylation'),
    ('AKT1', 'ESR1', 'phosphorylation'),
    ('ERK1', 'HIF1A', 'stabilization'),
    ('mTORC1', 'HIF1A', 'translation'),

    # Phenotypic Outcomes
    ('MYC', 'cell_proliferation', 'promotion'),
    ('AKT1', 'cell_survival', 'promotion'),
    ('ERK1', 'migration', 'promotion'),
    ('HIF1A', 'angiogenesis', 'promotion'),
]

# Add edges with random weights
for source, target, interaction in mapk_edges:
    G_mapk.add_edge(source, target,
                    weight=random.uniform(0.5, 2.0),
                    interaction=interaction)

# Add 150 random edges to increase complexity
for _ in range(150):
    src, tgt = random.sample(mapk_nodes, 2)
    if not G_mapk.has_edge(src, tgt):
        G_mapk.add_edge(src, tgt,
                        weight=random.uniform(0.1, 0.5),
                        interaction='random_crosstalk')

# ==============================================
# NETWORK VISUALIZATION
# ==============================================
def visualize_network(G, title, filename):
    plt.figure(figsize=(25, 25))
    pos = nx.spring_layout(G, k=0.4, seed=42)

    # Color nodes by function
    if G == G_cell_cycle:
        node_colors = [
            'red' if 'Cyclin' in node else
            'orange' if 'CDK' in node else
            'lime' if node in ['p21', 'p27', 'p16'] else
            'skyblue' if node in ['ATM', 'ATR', 'Chk1', 'Chk2', 'p53'] else
            'purple' if node in ['G1_phase', 'S_phase', 'G2_phase', 'M_phase'] else
            'lightgray' for node in G.nodes()
        ]
    else:  # MAPK network
        node_colors = [
            'red' if node in ['EGFR', 'HER2', 'IGF1R'] else
            'orange' if node in ['RAS', 'RAF1', 'MEK1', 'ERK1'] else
            'lime' if node in ['PI3K', 'AKT1', 'mTORC1'] else
            'skyblue' if node in ['DUSP1', 'SPRY2', 'PTEN'] else
            'purple' if node in ['cell_proliferation', 'cell_survival', 'angiogenesis'] else
            'lightgray' for node in G.nodes()
        ]

    nx.draw(G, pos, with_labels=True, node_size=800,
            node_color=node_colors, font_size=8,
            edge_color='gray', width=0.7, arrowsize=15)
    plt.title(title, size=20)
    plt.tight_layout()
    plt.savefig(filename, dpi=300)
    plt.show()

visualize_network(G_cell_cycle,
                 "Expanded Cell Cycle Network with Cross-Talk",
                 "expanded_cell_cycle_network.png")

visualize_network(G_mapk,
                 "Expanded MAPK Signaling Network with Cross-Talk",
                 "expanded_mapk_network.png")

# ============================================
# NETWORK ANALYSIS
# ============================================
def analyze_network(G, name):
    print(f"\n{name} Network Stats:")
    print(f"Nodes: {G.number_of_nodes()}, Edges: {G.number_of_edges()}")
    print(f"Avg Degree: {sum(dict(G.degree()).values()) / len(G):.2f}")

    if nx.is_weakly_connected(G):
        print(f"Diameter: {nx.diameter(G.to_undirected())}")
    else:
        print("Contains disconnected components")

    print(f"Clustering: {nx.average_clustering(G.to_undirected()):.3f}")

analyze_network(G_cell_cycle, "Cell Cycle")
analyze_network(G_mapk, "MAPK Signaling")
```

Listing A.14: Python implementation for network construction and analysis

Supplementary Materials

Complete Python Implementation

Below is the complete Python code used for network refinement, analysis, and visualization:

```
import networkx as nx
import matplotlib.pyplot as plt
from matplotlib.lines import Line2D
import pandas as pd
import numpy as np

# ======================
# CONFIGURATION
# ======================
# Color schemes
NODE_COLORS = {
    # Cell Cycle
```

```python
    'Cyclin D': '#FF7F0E', 'CDK4/6': '#FF7F0E',
    'Rb': '#9467BD', 'E2F': '#9467BD',
    'p53': '#2CA02C', 'p21': '#2CA02C',
    # MAPK
    'EGFR': '#1F77B4', 'RAS': '#1F77B4',
    'MEK': '#1F77B4', 'ERK': '#1F77B4'
}

EDGE_COLORS = {
    'phosphorylation': '#1F77B4',
    'activation': '#2CA02C',
    'inhibition': '#D62728',
    'complex': '#FF7F0E',
    'default': '#7F7F7F'
}

# ======================
# NETWORK LOADING
# ======================
def load_initial_network(filepath):
    """Load initial network from GEXF file"""
    return nx.read_gexf(filepath)

# ======================
# FILTERING FUNCTIONS
# ======================
def filter_by_essential(G, essential_nodes):
    """Filter network to only essential nodes"""
    filtered = nx.DiGraph()
    filtered.add_nodes_from(essential_nodes)
    for u, v, data in G.edges(data=True):
        if u in essential_nodes and v in essential_nodes:
            filtered.add_edge(u, v, **data)
    return filtered

def validate_edges(G, reference_db):
    """Validate edges against reference database"""
    validated = nx.DiGraph()
    validated.add_nodes_from(G.nodes())

    for u, v, data in G.edges(data=True):
        if (u, v) in reference_db:
            validated.add_edge(u, v, **{**data, **reference_db[(u, v)]})
    return validated

def merge_redundant_nodes(G, node_groups):
    """Merge similar/redundant nodes"""
    for group in node_groups:
        representative = group[0]
        for node in group[1:]:
            G = nx.contracted_nodes(G, representative, node)
    return G

# ======================
# ANALYSIS FUNCTIONS
# ======================
def calculate_network_stats(G):
    """Calculate key network statistics"""
    stats = {
        'nodes': G.number_of_nodes(),
        'edges': G.number_of_edges(),
        'avg_degree': sum(dict(G.degree()).values()) / len(G),
        'diameter': nx.diameter(G.to_undirected()) if nx.is_connected(G.to_undirected())
     else None,
        'clustering': nx.average_clustering(G.to_undirected())
    }
    return stats

def identify_key_nodes(G, top_n=5):
    """Identify top central nodes using multiple measures"""
    centrality = {
        'degree': nx.degree_centrality(G),
        'betweenness': nx.betweenness_centrality(G),
        ’pagerank’: nx.pagerank(G)
    }
    key_nodes = {}
    for measure, scores in centrality.items():
        sorted_nodes = sorted(scores.items(), key=lambda x: x[1], reverse=True)[:top_n]
        key_nodes[measure] = [node for node, score in sorted_nodes]
    return key_nodes

# ======================
# VISUALIZATION FUNCTIONS
# ======================
def visualize_network(G, title, filename, figsize=(12, 10)):
    """Generate publication-quality network visualization"""
    plt.figure(figsize=figsize)

    # Layout
    pos = nx.spring_layout(G, k=0.8, seed=42)

    # Node styling
    node_colors = [NODE_COLORS.get(node, ’#7F7F7F’) for node in G.nodes()]

    # Edge styling
    edge_colors = []
    edge_styles = []
    for _, _, data in G.edges(data=True):
        edge_colors.append(EDGE_COLORS.get(data.get(’type’, ’default’), ’#7F7F7F’))
        edge_styles.append(’-’ if data.get(’type’) != ’inhibition’ else ’--’)

    # Draw network
    nx.draw_networkx_nodes(G, pos, node_color=node_colors, node_size=800)
    nx.draw_networkx_labels(G, pos, font_size=8)

    for i, (u, v, data) in enumerate(G.edges(data=True)):
        nx.draw_networkx_edges(G, pos, edgelist=[(u, v)],
                              edge_color=edge_colors[i],
                              style=edge_styles[i],
                              width=1.5,
                              arrowsize=20,
                              arrowstyle=’->’)

    # Create legend
    legend_elements = [
        Line2D([0], [0], color=color, lw=2, label=label)
        for label, color in EDGE_COLORS.items()
    ]
    plt.legend(handles=legend_elements, loc=’upper right’)

    plt.title(title, fontsize=14)
    plt.tight_layout()
    plt.savefig(filename, dpi=300, bbox_inches=’tight’)
    plt.close()

# ======================
# MAIN EXECUTION
# ======================
if __name__ == "__main__":
    # Load initial networks
    cc_initial = load_initial_network(’cell_cycle_initial.gexf’)
    mapk_initial = load_initial_network(’mapk_initial.gexf’)

    # Define essential components
    cc_essential = [’Cyclin D’, ’CDK4/6’, ’Rb’, ’E2F’, ’p53’, ’p21’]
    mapk_essential = [’EGFR’, ’RAS’, ’MEK’, ’ERK’, ’PI3K’, ’AKT’]

    # Refine networks
    cc_refined = filter_by_essential(cc_initial, cc_essential)
    mapk_refined = filter_by_essential(mapk_initial, mapk_essential)

    # Merge redundant nodes
    cc_groups = [[’CDK4’, ’CDK6’, ’CDK4/6’]]
    mapk_groups = [[’ERK1’, ’ERK2’, ’ERK’], [’MEK1’, ’MEK2’, ’MEK’]]

    cc_refined = merge_redundant_nodes(cc_refined, cc_groups)
    mapk_refined = merge_redundant_nodes(mapk_refined, mapk_groups)

    # Annotate interactions
    interaction_db = {
        # Cell Cycle interactions
        ('Cyclin D', 'CDK4/6'): {'type': 'complex'},
        ('Cyclin D/CDK4/6', 'Rb'): {'type': 'phosphorylation'},
        # MAPK interactions
        ('EGFR', 'RAS'): {'type': 'activation'},
        ('RAS', 'MEK'): {'type': 'phosphorylation'}
    }

    cc_refined = validate_edges(cc_refined, interaction_db)
    mapk_refined = validate_edges(mapk_refined, interaction_db)

    # Analyze networks
    cc_stats = calculate_network_stats(cc_refined)
    mapk_stats = calculate_network_stats(mapk_refined)

    # Visualize networks
    visualize_network(cc_refined,
                    "Refined Cell Cycle Network",
                    "cell_cycle_refined.png")

    visualize_network(mapk_refined,
                    "Refined MAPK Network",
                    "mapk_refined.png")

    # Save final networks
    nx.write_gexf(cc_refined, "cell_cycle_final.gexf")
    nx.write_gexf(mapk_refined, "mapk_final.gexf")
```

Listing A.15: Complete network analysis pipeline

A.3 Code for Network Analysis and Simulation

```
import networkx as nx
import numpy as np
import matplotlib.pyplot as plt
import random

# Create Cell Cycle Pathway Network
cell_cycle_edges = [
    ('Cyclin D', 'CDK4/6', {'weight': 1}),
    ('Cyclin E', 'CDK2', {'weight': 1}),
    ('Cyclin A', 'CDK2', {'weight': 1}),
    ('Cyclin B', 'CDK1', {'weight': 1}),
    ('Cyclin D/CDK4/6', 'Rb', {'weight': 1}),
    ('Rb', 'E2F', {'weight': 1}),
    ('Phosphorylated Rb', 'E2F', {'weight': 1}),
    ('E2F', 'DNA replication genes', {'weight': 1}),
    ('DNA damage', 'ATM/ATR', {'weight': 1}),
    ('ATM/ATR', 'CHK1/CHK2', {'weight': 1}),
    ('CHK1/CHK2', 'CDK1', {'weight': 1}),
    ('CHK1/CHK2', 'CDK2', {'weight': 1}),
    ('ATM/ATR', 'p53', {'weight': 1}),
    ('p53', 'p21', {'weight': 1}),
    ('p21', 'CDK4/6', {'weight': 1}),
    ('p21', 'CDK2', {'weight': 1}),
    ('Spindle checkpoint components', 'Anaphase progression', {'weight': 1}),
    ('Mitotic Exit Network', 'Exit from Mitosis', {'weight': 1}),
    ('Growth factors', 'Cyclin D', {'weight': 1})
]

# Create MAPK Pathway Network
mapk_edges = [
    ('EGFR', 'RAS', {'weight': 1}),
    ('RAS', 'MEK', {'weight': 1}),
    ('MEK', 'ERK', {'weight': 1}),
```

```python
    ('ERK', 'RSK', {'weight': 1}),
    ('RSK', 'MYC', {'weight': 1}),
    ('EGFR', 'PI3K', {'weight': 1}),
    ('PI3K', 'AKT', {'weight': 1}),
    ('AKT', 'mTOR', {'weight': 1}),
    ('mTOR', 'p70S6K', {'weight': 1}),
    ('ERK', 'CREB', {'weight': 1}),
    ('ERK', 'ELK1', {'weight': 1}),
    ('ERK', 'RSK', {'weight': 1}),
    ('JNK', 'c-Jun', {'weight': 1}),
    ('p38', 'ATF2', {'weight': 1}),
    ('Ras', 'NF-kB', {'weight': 1}),
]

# Create Graphs for Both Pathways
G_cell_cycle = nx.DiGraph()
G_mapk = nx.DiGraph()

# Add edges to each network
G_cell_cycle.add_edges_from(cell_cycle_edges)
G_mapk.add_edges_from(mapk_edges)

# Random Walk Function
def random_walk(G, start_node, num_steps):
    current_node = start_node
    visiting_counts = {node: 0 for node in G.nodes()}

    for _ in range(num_steps):
        visiting_counts[current_node] += 1
        neighbors = list(G.neighbors(current_node))
        if not neighbors:
            break
        weights = [G[current_node][neighbor]['weight'] for neighbor in neighbors]
        current_node = np.random.choice(neighbors, p=np.array(weights) / sum(weights))

    return visiting_counts

# Parameters for Random Walks
num_walks = 1000
num_steps = 50
start_node_cell_cycle = 'Cyclin D'
start_node_mapk = 'EGFR'

# Perform Random Walks
total_visiting_counts_cell_cycle = {node: 0 for node in G_cell_cycle.nodes()}
total_visiting_counts_mapk = {node: 0 for node in G_mapk.nodes()}

# Perform random walks on both networks
for _ in range(num_walks):
    counts_cell_cycle = random_walk(G_cell_cycle, start_node_cell_cycle, num_steps)
    counts_mapk = random_walk(G_mapk, start_node_mapk, num_steps)

    for node in total_visiting_counts_cell_cycle:
        total_visiting_counts_cell_cycle[node] += counts_cell_cycle[node]

    for node in total_visiting_counts_mapk:
        total_visiting_counts_mapk[node] += counts_mapk[node]

# Normalize visit counts
for node in total_visiting_counts_cell_cycle:
    total_visiting_counts_cell_cycle[node] /= num_walks

for node in total_visiting_counts_mapk:
    total_visiting_counts_mapk[node] /= num_walks

# Plot Visit Counts for both pathways
def plot_visit_counts(visiting_counts, title):
    plt.figure(figsize=(12, 6))
    nodes = list(visiting_counts.keys())
    counts = list(visiting_counts.values())
    plt.bar(nodes, counts, color='royalblue')
    plt.xlabel('Nodes', fontsize=12)
    plt.ylabel('Normalized Visit Count', fontsize=12)
    plt.title(title, fontsize=14)
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()

plot_visit_counts(total_visiting_counts_cell_cycle, 'Normalized Visit Counts - Cell Cycle Pathway')
plot_visit_counts(total_visiting_counts_mapk, 'Normalized Visit Counts - MAPK Signaling Pathway')
```

Listing A.16: Python Code for Pathway Analysis and Random Walks