Ümit Bulut VISUALIZATION OF INTERSECTING SETS IN BIOLOGICAL NETWORKS AND FILTERING BASED ON THEIR TOPOLOGICAL PROPERTIES AGU 2020

VISUALIZATION OF INTERSECTING SETS IN BIOLOGICAL NETWORKS AND FILTERING BASED ON THEIR TOPOLOGICAL PROPERTIES

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER SCIENCE
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTERS IN SCIENCE

By
Ümit Bulut
May 2020

VISUALIZATION OF INTERSECTING SETS IN BIOLOGICAL NETWORKS AND FILTERING BASED ON THEIR TOPOLOGICAL PROPERTIES

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER SCIENCE
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTERS IN SCIENCE

By
Ümit Bulut
May 2020

SCIENTIFIC ETHICS COMPLIANCE

I hereby declare that all information in this document has been obtained in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name-Surname: Ümit Bulut
Signature :

REGULATORY COMPLIANCE

M.Sc. thesis titled “Visualization Of Intersecting Sets In Biological Networks And Filtering Based On Their Topological Properties” has been prepared in accordance with the Thesis Writing Guidelines of the Abdullah Gül University, Graduate School of Engineering & Science.

Prepared By	Advisor
Ümit Bulut	Asst. Prof. Burcu Bakır Güngör
Signature	Signature

Head of the Electrical and Computer Science Program

Prof. Dr. V. Çağrı Güngör

Signature

ACCEPTANCE AND APPROVAL

M.Sc. thesis titled “Visualization Of Intersecting Sets In Biological Networks And Filtering Based On Their Topological Properties” and prepared by Ümit Bulut has been accepted by the jury in the Electrical and Computer Science Graduate Program at Abdullah Gül University, Graduate School of Engineering & Science.

16 / 06 / 2020

(Thesis Defense Exam Date)

JURY:

Advisor : Asst. Prof. Burcu Bakır Güngör ……signature

Member : Asst. Prof. Özkan Ufuk Nalbantoğlu …… signature

Member : Dr. Ahmet Soran …… signature

APPROVAL:

The acceptance of this M.Sc. thesis has been approved by the decision of the Abdullah Gül University, Graduate School of Engineering & Science, Executive Board dated ….. /….. / …….. and numbered ……..……. .

…….. /…….. / ……..

(Date)

Graduate School Dean
Prof. Dr. İrfan Alan,
Signature

ABSTRACT

VISUALIZATION OF INTERSECTING SETS IN BIOLOGICAL NETWORKS AND FILTERING BASED ON THEIR TOPOLOGICAL PROPERTIES

Ümit Bulut
MSc in Electrical and Computer Science
Supervisor: Asst. Prof. Burcu Bakır Güngör

May 2020

With the rapid development of computer sciences and data processing technologies, bioinformatics became one of the rising multidisciplinary fields in this century. In this thesis, we aim to introduce a new perspective on bioinformatics data analysis via developing a new visualization software called BioNetVis. Our tool has ability to visualize intersecting sets in the biological networks, especially in the protein-protein interaction networks; and to filter based on graph topological measures. BioNetVis, is developed with the latest versions of state-of-the-art frameworks and programming libraries for processing data as fast as possible with higher efficiency. The main goal of BioNetVis is to facilitate the analysis of intersecting biological datasets on biological networks. The proposed tool aims to serve to the researchers who are working in the field of drug repurposing, personalized medicine, diagnosis and treatment of rare diseases.

The project implementation is realized in the following three steps. Firstly, the biological data is mapped to a biological network and back-end development is performed. Secondly, a visualization is created based on the processed data in the back end with latest framework services. Thirdly, the back-end and front-end developments are connected and BioNetVis is made available to the researchers. We design BioNetVis in a modular fashion such that it is applicable to other types of networks and datasets and hence, it could be used in other domains to visualize intersecting sets in networks and filter based on graph topological properties. Lastly, we present a use case scenario to explain the features of BioNetVis.

Keywords: Bioinformatics, Data Visualization, Intersecting Data Sets, Drug Repurposing, Personalized Medicine

ÖZET

BİYOLOJİK AĞLARDA KESİŞEN KÜMELERİN GÖRSELLEŞTİRİLMESİ VE TOPOLOJİK ÖZELLİKLERİNE GÖRE FİLTRELENMESİ

Ümit Bulut
Elektrik ve Bilgisayar Mühendisliği Bölümü, Yüksek Lisans
Tez Yöneticisi: Dr. Öğr. Üyesi Burcu Bakır Güngör

Mayıs 2020

Bilgisayar bilimleri ve veri işleme teknolojilerinin hızlı gelişimiyle beraber, biyoenformatik alanı, bu yüzyılın popülerliği artan disiplinlerarası çalışma alanlarından birisi haline gelmiştir. Bu tez çalışmasında, BioNetVis isimli yeni bir göreslleştirme aracı geliştirerek, biyoenformatik very analizine yeni bir perspektif getirmeyi amaçladık. Geliştirdiğimiz araç, keşisen kümeleri biyolojik ağlar, özellikle protein-protein etkileşim ağları üzerinde görselleştirme ve ağ topolojik özelliklerine göre filtreleme yeteneğine sahiptir.

BioNetVis, verileri olabildiğince hızlı ve verimli bir şekilde işlemek için son teknoloji frameworkler ve programlama kütüphaneleri ile geliştirilmiştir. BioNetVis'in ana amacı kesişen biyolojik verileri, biyolojik ağlar üzerinde analiz etmeyi kolaylaştırmaktır. Sunulan araç, ilaçların yeniden konumlandırılması, kişiselleştirilmiş tıp, nadir hastalıkların teşhis ve tedavisi konularında çalışan araştırmacılara hizmet etmeyi amaçlar. Proje şu üç adımda gerçekleştirilmiştir. İlk olarak, biyolojik veri biyolojik ağa haritalanmış, ve back-end kısmı geliştirilmiştir. İkinci olarak, back-end kısmında işlenmiş veriler, kullanıcılar için kodlanan arayüz ile görselleştirilmiştir. Üçüncü olarak, front-end ve back-end kısımları birleştirilmiş ve araştırmacılar için kullanılabilir hale getirilmiştir. BioNetVis'i diğer ağlara ve diğer veri kümelerine kolayca uyarlanabilsin diye modüler olarak tasarladık. Böylece, BioNetVis diğer alanlarda kesişen veri kümelerinin ağda görsellenmesi, ve ağ topolojik özelliklerine göre filtrelenmesi amacıyla kullanılabilir. Son olarak, BioNetVis'in sunduğu işlevleri açıklamak için, bir kullanım uygulaması ile beraber sunduk.

Anahtar kelimeler: Biyoenformatik, Veri Görsellenmesi, Kesişen Veri Kümeleri, İlaçların Yeniden Konumlandırılması, Kişiselleştirilmiş Tıp

Preface

In this short, one page, preface you will find my other works unrelated to this thesis but has been done during my master's education. Before preparing such adventures and enjoyable thesis, I have had opportunities to discover many interesting areas such as computer vision, robotics and machine learning. After some debate and brainstorming I have decided to work on an implementation project that will help people in the medical research area mostly, since healthcare is very important as we can see during these unfortunate times.

Acknowledgements

All of these acknowledgments are sincere and comes from deepest place of my heart.
First of all, I want to thank my brilliant, dedicated and diligent advisor Asst. Prof. Burcu Bakır Güngör, whom was the one believed in me and helped me get through this adventure. I am humbled for working with her and always take her as a role model while walking academic life. I have never seen a human who can prepares lectures with so many information and knowledge, let all expense classes fully.
I want to thank my friends Adam Rizvi Thahir and Hasan A.A. Alsulaiman who guide me while I was learning new coding languages, listened to me while I was stuck in problems and listen my ideas.
Also I want to thank my dearest friend Adem Emre MD, for all the help and motivation during my masters. He has been there for my good times and hard times. I am very lucky to have a friend like him.
Additionally, I want to thank and congratulate myself by finishing this thesis. It was not easy to complete this journey. I know there has been times it felt like this road won't finish at all. So good work to me.
Lastly, I want to give a huge thanks to my family especially my parents for their incredible support in many ways that I can't express with words at all. Thank you for your patience, time and support. Without their love and fate in me, this thesis would not exist. So I can't thank enough for them. They are the unbreakable pillars of my life and success. I wish my mother could see this work, but life works in mysterious ways. I know she knew I would have completed it. So mom, here I am, completed my work. Thanks to all of my family members.

Table of Contents

List of Figures

List of Tables

To my beloved parents...
Ayfer Su and Ebumuhsin

Chapter 1

Introduction

In this modern world; with the increasing computing power and connection around researchers causes a growth in human knowledge rapidly. On the other hand, this knowledge needs to be validate, corrected then share with the world especially in the medical related fields. Currently, in a post-truth era circulated knowledge and news needed to be examine very carefully.

In this work, we aimed to help researchers, most importantly many of whom work in the biological – chemical related fields and subfields where try to decode human body. A visual interactive interface called Protein Interaction Visualization Tool, from now referred as BioNetVis as short, is developed in order to understand relations between proteins and their topological properties. Moreover, BioNetVis has a powerful capability to show this proteins relationships in Human Genome, Protein - Protein Interaction Network whether these proteins used arbitrary to users wishes, discovered in experiments in the laboratories or found in collected datasets.

As name suggests visualization in BioNetVis, enables cognitive abilities to gain much more insights in these especially specialized areas which requires much prior knowledge in order to understand how proteins interacts each other. With that being said, parallel to previous aim, gaining insight, other properties of BioNetVis is that filtering proteins based on their occurrences, neighborhood properties and importance.

Filtering and visualization will open a new pathway to researchers, drug developers to discover personalized medicine, drug repurposing and examination of rare diseases.

Among many science field related to biology, it is necessary to analyze and compare data sets contains various information such as genes, proteins, enzymes, systems, organism and such. This analysis of datasets can be beneficial for many other science fields as well. For this purposes, data sets and their intersections, unions often displayed with Venn diagrams which they are accepted and used widely. [1] One latest example for use of such diagrams can be seen in the article for using machine learning approach to classify accurate detection of copy number variants (CNVs) from exome sequencing where a Venn diagram shows the accuracy of the concordance profiles of the CNVs before and after the classification. [2] Venn diagrams are could be helpful as a way to give a basic exploration and understanding of data but regularly this diagrams provides solely static points of the data up to four sets. Even though Venn diagrams may have built for more than four sets, the layout of the visualization difficulty increases exponentially. In example, any user can create symmetric circles to use three set of diagrams but for more than four sets highlighting and understanding intersections, excluded areas are relatively challenging. Same challenge follows with ellipses as well. After five sets of data, it becomes harder to visualize.

Interdisciplinary work is inevitable with the advancements in the research areas. Scientists, researchers need to work together in multidisciplinary way in order to enhance human knowledge. Despite the fact that working together is necessary, finding a partner to work with or finding a participant that interested in that specific area can be hard to tackle. But we believe these kind of limitations should not hold back researchers especially in the study fields such as bioinformatics and computer sciences. Many biology, medical related people have limited information about computer science and coding for a specific task might be time consuming for them. Not so surprisingly, similar problem occurs among in computer scientist. Even though they have developing skills to code what it necessary, understanding biology related problem can be complex. On the other hand, other fields who does the knowledge discovery, investigate, from the big data, finding repetitions, occurrences also, creating a network from scratch is problematic. BioNetVis aims to make a bridge between these two problems and

make it an easy way. In addition to that, currently most of the graph tools are either outdated in terms of look and performance or very inefficient, often requires prior knowledge.

This problem can be solved in a very simple way. Problem is that bioinformaticians needs a user interface (UI) to wrangle with their data that is not too complicated to understand, not time consuming to learn prior knowledge and tools functions. So interface should be simple yet can be done complex tasks in behind. Creating such UI itself is a challenging task, we will mention this problem in the materials and methods section. To understand how much similarity, the data has and also data's topological properties in a network can be solved with proposed UI. We are proposing an implementation project that will allows users to understand their data much more deliberately.

There are some old approaches in order to see similarities in the data with Venn diagrams such as online web pages and other package programs but problem with these approaches that they all lack either from the performance or usability perspective. In chapter 2, there is some literature review related to the online tools that have been using Venn diagrams with their features which often seems as basic way to a high level publication.

We combine visualization considering artistic and academic looks with the computer science and bioinformatics. In our proposed approach we are making it much more compact with the algorithm that we have been developed so while we have maximum usability in terms of user experience with easiness, on the other hand we do not have to compromise from the performance as well. Our visualization approach comes with filtering which is many web based tools lack of. Also having the performance without all the installations and required knowledge to learn how to use complicated package programs make the BioNetVis obvious choice while trying to understand Protein-Protein Interactions (PPI) or any other data that has similarities and needs to be visualized.

We believe in many aspects with the developments in the computer science should have a reflection to the other fields as well. This includes creating tools that enable researchers to improve their research much more efficiently with the state-of-the-

art tools and implementations. Currently, our proposed tool is runs on the cutting edge technology in terms of used libraries, frameworks, languages and visualization as well. It has an understandable, easy to use minimalistic user interface so that instead of trying to comprehend complex bioinformatics desktop apps or dealing with first-grade level visualization with outdated tools, users actually can be focused on their work not the looks.

Since this thesis is an implementation project, there is no way to get an objective result that will justify one program to another. Instead, we prepared a use case for the thesis and demonstrated it on the current working system. Despite the advantages, we need to highlight that BioNetVis has limitations in bioinformatics, since it focuses mainly one job and that is visualization on Networks and clusters. BioNetVis is not aiming replacing the current bioinformatics tools or make a claim that it is better than other tools. Instead we are presenting our simplistic and powerful approach for one part in the incredibly knowledge needs to be discover in bioinformatics and other research fields as well.

Neither visualization of protein visualization and finding occurrences in different clusters are new. There has been intensive study about proteins, visualization of them, their specialization, why and how they work and relations have been made. But we have not found any study that been made about protein-protein interactions and visualization of this PPI in a human network as a tool. We can say this is a new approach or alternative to the current studies that are most likely needs to improved.

Of course once you have developed a project or thesis a question that comes to mind is “What is the contribution of this work?” We contribute to the area of bioinformatics tools in terms of adapting new technologies in computer sciences. Also BioNetVis hopes to be achieve a tool that every bioinformaticians had used for their works. Hence, BioNetVis’ audience is broad and not limited to one specific subfield. It will continue to update itself in time.

Chapter 2

Theoretical Background

Having an extreme amount of information is a well-known fact of the information age because of the achievements in computer science, in terms of increasing computing power and the amount of data storage capacity and thus data is produced in extreme amounts.

While we can collect and store the data faster and easily, our ability to analyze that large amount of data comparatively becomes slower. Analyzing these kinds of generally disorganized, hard to understand big data is decisive in many domains. It is an important task for the people in charge such as business analysts, decision-makers to extract and use the appropriate information regarding flowing data that keeps coming regularly.

A number of software tools are now being used in order to help analysts to organize their data, generate overviews and explore the area of knowledge to gather useful information.

Many of the tools often developed based on the specific needs of the developers may be perceived as to complex for daily users. On the other side algorithms and connections between the programmer and the data planned as according to purpose of programmer, while algorithms appear generalized too often for the user. And because of that massive time and resource, resources are often lost due to lack of proper interaction between databases and the users.

Visual computing seeks to bridge this gap through the use of increasingly sophisticated analytical methods. The fundamental idea of representing data in the visual domain is to allow humans to interact with the information directly. Therefore, it will be much easier to gain insight and draw conclusions much more efficiently. People may use visual analytics tools and techniques to synthesize

information and derive insight from massive, dynamic, and often conflicting data by providing timely, defensible, and understandable assessments. [3]

Visual analysis research aims to transform the abundance of data into an opportunity, hence knowledge. In order to make efficient decision-making in absolute critical situations, decision-makers should be allowed to examine a massive, multi-dimensional, time-differentiated information in order to see the patterns in the big data. Visual analytics have a particular benefit in that decision-makers may focus on the analysis method their perceptual and cognitive abilities and visual capacity while enabling them to use advanced computer technologies to improve the discovery process. In this thesis, our aim to suggest one alternative to enable such opportunities with a tool that is developed specifically for one task on the visualization of bioinformatics networks and based on their topological specialties.

There are many visualization tools that are developed for general purposes. These tools differ based on their primary purposes and also their working environment. While some of them are only to visualize the data, some of them enable interactive editing of the graph content. [4] All of these tools such as web-based tools, desktop applications and command-line based use different algorithms specific to their needs. Table 2.1 shows a comparison of some of the popular tools and their purposes.

Tools	**Graph Editing**	**Program Assistance**	**Layout**	**Customizability**	**Availability**
Gephi	Yes	Yes[1]	Specified[2]	High[3]	Open Source[4]
NetworkX	No	No	Editable[5]	Low[6]	Open Source
Cytoscape	Yes	Yes	Specified	Medium	Open Source
Graphviz	Yes	No	Specified	Medium	Open Source
upsetR	No	No	Specified	High	Open Source
Venny	No	No	Specified	Low	Available[7]
Venture	No	No	Specified	Low	Private[8]

Vennplex	No	No	Specified	Low	Private
BioNetVis	**Yes**	**No**	**Editable[5]**	**Medium**	**Open Source**

1 Knowledge base, forums and online groups are indicated as support.
2 Supplementary layouts cannot adapt to program easily
3 Based on programs built-in customization preferences
4 Source code publicly available and it is free software
5 Layout can be changed via editing source code
6 Restricted to simple node and edge visual properties
7 Program is publicly available, source code can be reached via web browsers however, developer does not shared source code available
8 Free software, source code is not publicly available

Table 2.1: Comparison between BioNetVis to other popular web-based visualization software

Understanding complex data and make it out of meaningful often come as a challenge. Hence, most of the time researchers, academics uses visualization in order to activate cognitive parts of human brain. In a result of that there are many ways to approach visualization topic. Since we cannot cover all of the topics about visualization, we're going to explain visualization related to bioinformatics, especially network visualization and protein-protein interactions visualizations. This includes data types that have been used in this thesis work, libraries, and biological background of the diseases and data collection. Also, we will present other tools to visualize their specific task in bioinformatics area for understanding there are many complexities in the matter subject and show that one program cannot do all of the task related to bioinformatics as well as a person could expect. In below, we start with the importance of the data and its' processing parts. Then we will dive in bioinformatics, the areas related to bioinformatics and the data itself.

2.1 Bioinformatics, It's Scope and Data

Among many natural sciences, biology is one of the main pillars in order to understand life and world. Build-up of this information throughout to history take a new turn with new inventions such as computers and world-wide-web. In this age we can express biological information as molecular sequence of data, experimented content of genome and gene product analysis. [5]

“Being an interdisciplinary branch of the life sciences, bioinformatics targets to develop methodology and analysis tools to explore large volumes of biological data, helping to store, organize, systematize, annotate, visualize, query, mine, understand, and interpret complex data volumes. It uses conventional, modern computer science and cloud computing, statistics, and mathematics, as well as pattern recognition, reconstruction, machine learning, simulation and iterative approaches, and molecular modeling/folding algorithms.” [5], [6].

“The emergence and advances of the bioinformatics field, however, are tightly associated with the computerized programming and software developments needed for the handling and structural and functional analysis of large volumes of molecular sequences of DNA, RNA, proteins, and metabolites.” [7]

Following chapters, we will expand our description about bioinformatics and its applications in order to understand this thesis. In chapter 2.1.1 we will discuss about bioinformatics and its scope. Following chapter 2.1.2, explains bioinformatics data, importance of verified data, sharing of it, it’s collection and how the data is related to this project.

2.1.1 What is Bioinformatics and Its Scope

“Bioinformatics, a hybrid science that links biological data with techniques for information storage, distribution, and analysis to support multiple areas of scientific research, including biomedicine. Bioinformatics is fed by high-throughput data-generating experiments, including genomic sequence determinations and measurements of gene expression patterns. Database projects

curate and annotate the data and then distribute it via the World Wide Web. Mining these data leads to scientific discoveries and to the identification of new clinical applications. In the field of medicine in particular, a number of important applications for bioinformatics have been discovered. For example, it is used to identify correlations between gene sequences and diseases, to predict protein structures from amino acid sequences, to aid in the design of novel drugs, and to tailor treatments to individual patients based on their DNA sequences (pharmacogenomics).” [8]

“Common uses of bioinformatics include the identification of candidates genes and single nucleotide polymorphisms (SNPs). Often, such identification is made with the aim of better understanding the genetic basis of disease, unique adaptations, desirable properties (esp. in agricultural species), or differences between populations.” [6]

2.1.2 Bioinformatics Data

"In order to understand this thesis work, there are some prior knowledge related to bioinformatics needs to be learn. These are the subfields of the bioinformatics and main subjects that researchers work on. We will briefly define this subjects and will expand related parts such as proteomics for PPI and BioNetVis.

What is omic data?

“Omics informally refers to a field of study in biology ending in -omics, such as genomics, proteomics or metabolomics. Omics aims at the collective characterization and quantification of pools of biological molecules that translate into the structure, function, and dynamics of an organism or organisms.” [9] There are many fields that named with –omics additions. Some of them are: Genomics, Epigenomics, Proteomics, Glycomics, Transcriptomics, Metabolism. Even this fields have specialized subfields like cognitive genomics, comparative genomics etc. so that instead of searching needle in the haystack, researchers can reach needed information in vast natural sciences about human body. In this thesis work, developed visualization tool BioNetVis can be used many datasets and datatypes,

but especially for personalized medicine and drug discoveries, it uses -omics data, to be exact proteomic data. For future references and details following section we will explain some of the basics in bioinformatics and elaborate more on necessary subjects.

Deoxyribonucleic acid (DNA), is a molecule composed of two chains that coil around each other to form a double helix carrying genetic instructions for the development, functioning, growth and reproduction of all known organisms and many viruses. DNA and ribonucleic acid (RNA) are nucleic acids; alongside proteins, lipids and complex carbohydrates (polysaccharides), nucleic acids are one of the four major types of macromolecules that are essential for all known forms of life. [10]

DNA consists organic polymer of four different monomers. Each monomer is composed of a phosphate group, a single-ring sugar and one of four bases: A, T, C and G. The bases are planar single- or double-ring aromatic compounds. The monomers can form bonds between the sugar and the phosphate and form into a long polymer or strand of DNA. Such a single strand is irregular and if it consists of N monomers, there can be 4^N different combinations of bases, so the molecule carries 2^{2N} bits of information. [11] This basic formula itself reveals how complex the bioinformatics data can be. Genomics is an interdisciplinary field of biology focusing on the structure, function, evolution, mapping, and editing of genomes. A genome is an organism's complete set of DNA, including all of its genes. In contrast to genetics, which refers to the study of individual genes and their roles in inheritance, genomics aims at the collective characterization and quantification of all of an organism's genes, their interrelations and influence on the organism. [12], [13].

“Ribonucleic acid (RNA) is a polymeric molecule essential in various biological roles in coding, decoding, regulation and expression of genes. RNA and DNA are nucleic acids, and, along with lipids, proteins and carbohydrates, constitute the four major macromolecules essential for all known forms of life.” RNA typically is a single-stranded biopolymer. However, the presence of self-complementary sequences in the RNA strand leads to intrachain base-pairing and folding of the

ribonucleotide chain into complex structural forms consisting of bulges and helices. [14], [15] The transcriptome is the set of all RNA molecules in one cell or a population of cells. It is sometimes used to refer to all RNAs, or just mRNA, depending on the particular experiment. [16] Transcriptomics is the field studies RNAs and transcriptomes.

Proteins are vital parts of living organisms, with many functions hence to be able to understand proteins is important pathway to progress on human knowledge in medical applications and more. The study of proteins in large scale called proteomics. [17], [18]. “The proteome is the entire set of proteins that is produced or modified by an organism or system. Proteomics has enabled the identification of ever increasing numbers of protein. This varies with time and distinct requirements, or stresses, that a cell or organism undergoes.” [19] Proteomics is an interdisciplinary domain that has benefitted greatly from the genetic information of various genome projects, including the Human Genome Project (HGP). “The Human Genome Project (HGP) was an international scientific research project with the goal of determining the base pairs that make up human DNA, and of identifying and mapping all of the genes of the human genome from both a physical and a functional standpoint.” [20] It remains the world's largest collaborative biological project while writing this thesis in spring of 2020, besides an abnormal situation which is current pandemic crisis. [21] For mental issues of thesis writer, we skip crisis for now.

“One major development to come from the study of human genes and proteins has been the identification of potential new drugs for the treatment of disease. This relies on genome and proteome information to identify proteins associated with a disease, which computer software can then use as targets for new drugs. For example, if a certain protein is implicated in a disease, its 3D structure provides the information to design drugs to interfere with the action of the protein. A molecule that fits the active site of an enzyme, but cannot be released by the enzyme, inactivates the enzyme. This is the basis of new drug-discovery tools, which aim to find new drugs to inactivate proteins involved in disease. As genetic differences among individuals are found, researchers expect to use these

techniques to develop personalized drugs that are more effective for the individual.” [22]

“A metabolite is the intermediate end product of metabolism. The term metabolite is usually restricted to small molecules. Metabolites have various functions, including fuel, structure, signaling, stimulatory and inhibitory effects on enzymes, catalytic activity of their own (usually as a cofactor to an enzyme), defense, and interactions with other organisms (e.g. pigments, odorants, and pheromones). Metabolomics is the scientific study of chemical processes involving metabolites, the small molecule substrates, intermediates and products of metabolism. Specifically, metabolomics is the "systematic study of the unique chemical fingerprints that specific cellular processes leave behind", the study of their small-molecule metabolite profiles.” [23]–[25]

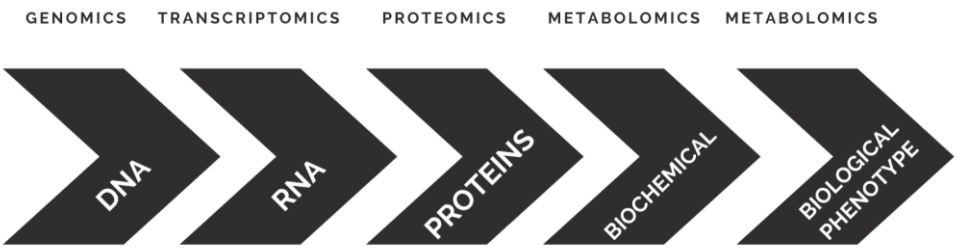

Figure 1: Central research areas of bioinformatics based on the organism and materials they focus on. [6]

2.1.3 Drug Repurposing

Biopharmaceutical industries’ one of the constant problem is despite of extreme amount of spending for research and development (R&D) output is always fallen short. This disadvantage forced many of drug developers to find creative solutions to improve productivity such as finding new uses or improve versions of existing drugs. [26] According to a report prepared by the Eastern Research Group (ERG), it takes between 10-15 years to develop a new drug. However, the success rate of developing a new molecular entity is only 2.01% on average. Additionally, based on Ashburn et. al. claims de novo approach (classical approach) takes 10-17 years to develop a new drug. [26]–[29]

Drug repurposing as known as drug repositioning, drug reprofiling or re-tasking is a method for discovering new uses for approved or investigational drugs that are outside of their initial usage means. [26], [30]

This pathway provides strong advantages in comparison to developing a new drug completely from the start. First and maybe the most importantly, chances of developing a failing project is lower since the repurposed drug has been already found. It may be in use or it is safe for preclinical models, early stage human trials hence it is unlikely to fail in terms of efficiency and resources perspective. Secondly, perhaps important as much as the first point, from our recent experiences during the world's pandemic, the time frame for drug development is reduced excessively because of drug's preclinical testing, safety appraisal and many cases formulation of the development is already have been completed. Additionally, because of first and second points, investment for the R&D process is decreased proportionally. That means, less investment will be needed although for many drug discovering cases this investment is relative both for time and money wise. [30], [31]

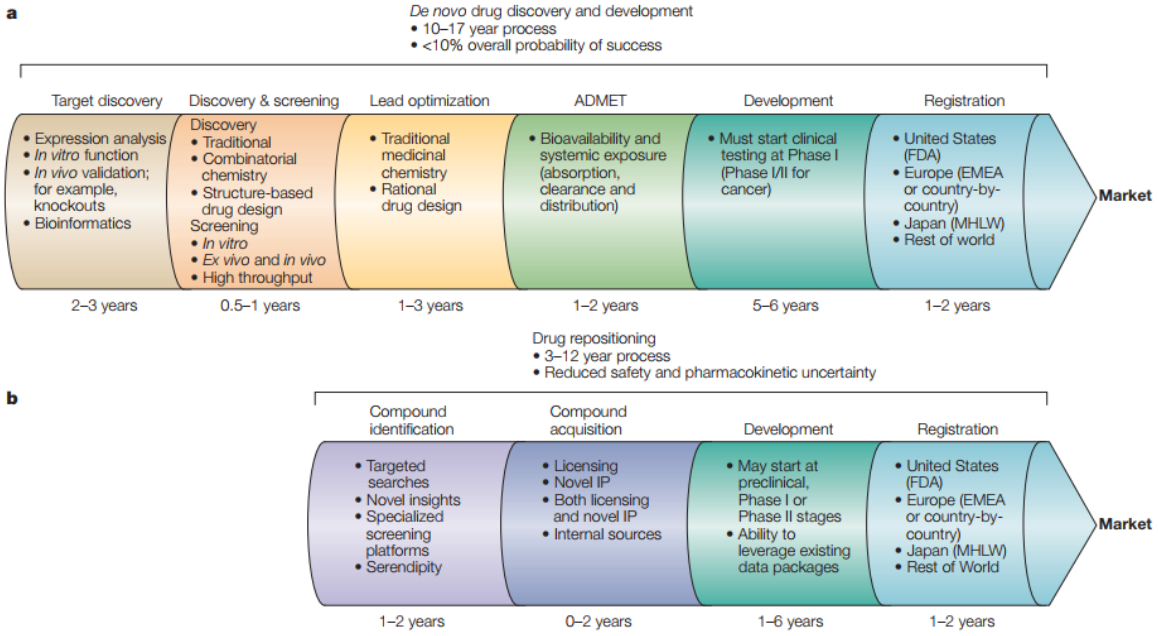

Figure 2: A comparison of traditional de novo drug discovery and development versus drug repositioning (taken from article) [26]

Moreover, there are several drug repurposing techniques such as computational drug repurposing. Like this thesis study's contribution, computational drug

repurposing or known as computer assisted drug repurposing is one of highlighted areas in the drug development processes. Especially with the increasing computing power and applications of machine learning techniques, processing extreme amount of data and speed factor makes computational drug repurposing a required technology of the future. [32]

A recent study about a pathophenotype of the SARS-CoV2 virus, widely known as COVID-19 (Coronavirus Disease 2019), which currently has not been found any drug or vaccine, shows the importance of computational drug repurposing. [33] Gysi et. al. shows that network based drug repurposing studies are much more promising than trying to find a drug from scratch. These drug repurposing studies are currently limited due to sparse domains of the area. BioNetVis aims to fill this gap over the years with growing features and abilities. For that reason, this study aims to be part of computational drug repurposing area.

To summarize the process of finding new uses outside the scope of the original medical indication for existing drugs is also known as redirecting, repurposing, repositioning and reprofiling. [26], [34] And this matter not only effects pharmaceutical industries, social aspects of drug repurposing is important as much as topic itself. Without a doubt societies approach to the pharmaceuticals and the companies will affect the culture, how we focus and regulate the industry itself. [35]. A simple use case for drug repurposing with BioNetVis will be shown in the following chapters. For furthermore reading please refer cited resources. [26]–[31], [34]–[41]

2.2 Various Approaches to Visualization Tools

There are many ways to approach data and its visualization tools. Many of them offers different specialties and utilities on separate platforms. For instance, some tools purpose only to work for one specific task, on the other hand some does more complicated work and requires prior knowledge and background for it. And the subject it not limited to only tools purposes but it’s design becomes one of the important factor while we asses. In light of this statements we can say there are many ways to develop a visualization tools. In this section we will take a look for

common tools that are already developed and offers different settings. You will find out that some tools are approach to handling biological data for one specific task only and because of it, these make them very simplistic and limited. On the other hand, complex tools are often cost time and money. On top of that this complex tools are most of the time not address required knowledge, it needs many variables in order to work hence in a way there are limitations for this complex tools as well and they do not answer user's needs.

When we wanted to approach visualization tool we decided to divide topic into the two parts. These are; web based visualization tools and package based visualizations tools which refers it requires and installation package and works as a desktop application. While we search at web based tools, instead diving into all of separate tools that does bioinformatics task for different purposes we will show tools that shows similar task which is; one: finding common strings and second showing them on a network.

2.3 Web Based Visualization Tools

Achievements in computer science effects every science field in many aspects. Especially in terms of data transfer, sharing and calculation. In result of that currently there are many bioinformatics tools that can be accessed via web. All of these web based tools has different purposes. Since this project is related to visualization in biological networks, we will cover visualization tools related to subject. Most of web-based bioinformatics visualization tools has one properties: finding common set of strings and visualize them in Venn Diagrams.

As mentioned in introduction, Venn Diagrams are beneficial while comparing sets. Diagrams are easy to use and interpret while number of sets are equal or less than four. There are several suggestions for more than four sets; Edwards et. Al. proposed using different shapes such as triangles, squares, ellipses and spheres. Edwards-Venn diagrams. [1] Nevertheless, with the increasing number of sets it becomes harder to interpret and visualize because of regions that needs to be calculated and created. In short: taking into account esthetic constraints and

human visual capacity pressures, more than four-sets and maximum seven-set Venn diagrams appear just too excessive. [42] BioNetVis offers the eliminate this limitations with implemented methods, without a network, it is much easier to interpret more than four sets. Although there are some desktop-based applications with various preferences like open-source availability or commercial purposes such as paid subscriptions or one-time-only license payments which we will cover at chapter 2.4, on the other hand with the help of increasing computation power and bandwidth speed, researchers can use some web-based tools as well. Following subsections in this chapter 2.3 we will look at some of the Venn Tools and their offerings.

Please note that, none of these web-based tools are capable of showing a cluster, set of strings, in a network. While BioNetVis offers showing similarity between sets, which elements in the strings are identical, also it shows clusters in the human genome network. On the other hand, this vennn diagrams can show only similarity, let alone visualize them in a network.

2.3.1 Venny

Venny is an interactive tool that built for comparing lists in venn diagram version. It can work online or you can save its .html file in order to work offline as well. Users can reach the website from Biognp's website. [43] Currently Venny version is 2.1 and users can reach version 1.0 as well. It is built in 2007 by Juan Carlos Oliveros whom works in BioinfoGP Service Centro Nacional de Biotecnología, (CNB-CSIC). Venny's biggest downside is limitations. It can compare and draw at most four-sets.

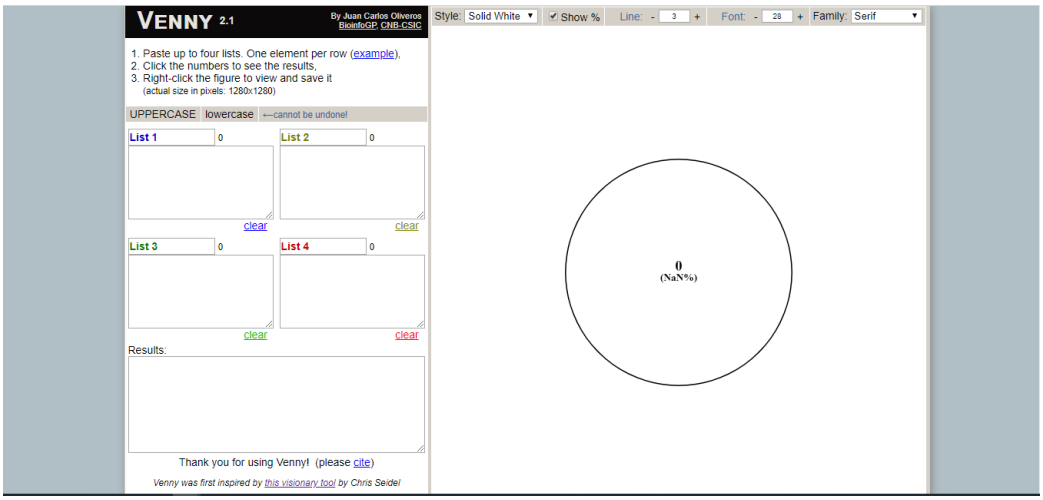

Figure 3: Venny 2.1 Webpage [43]

2.3.2 VennDiagram

VennDiagram, an R package that enables the automated generation of highly-customizable, high-resolution Venn diagrams with up to four sets and Euler diagrams with up to three sets. [44] Although it offers customization for diagrams which many online tools has, mentioned in chapter 2.3's, even though it works on a complier, with all the computing power from IDE, VennDiagram lacks performance since it offers four sets at the very utmost. Regrettably, the process of their command-line is not user-friendly as well.

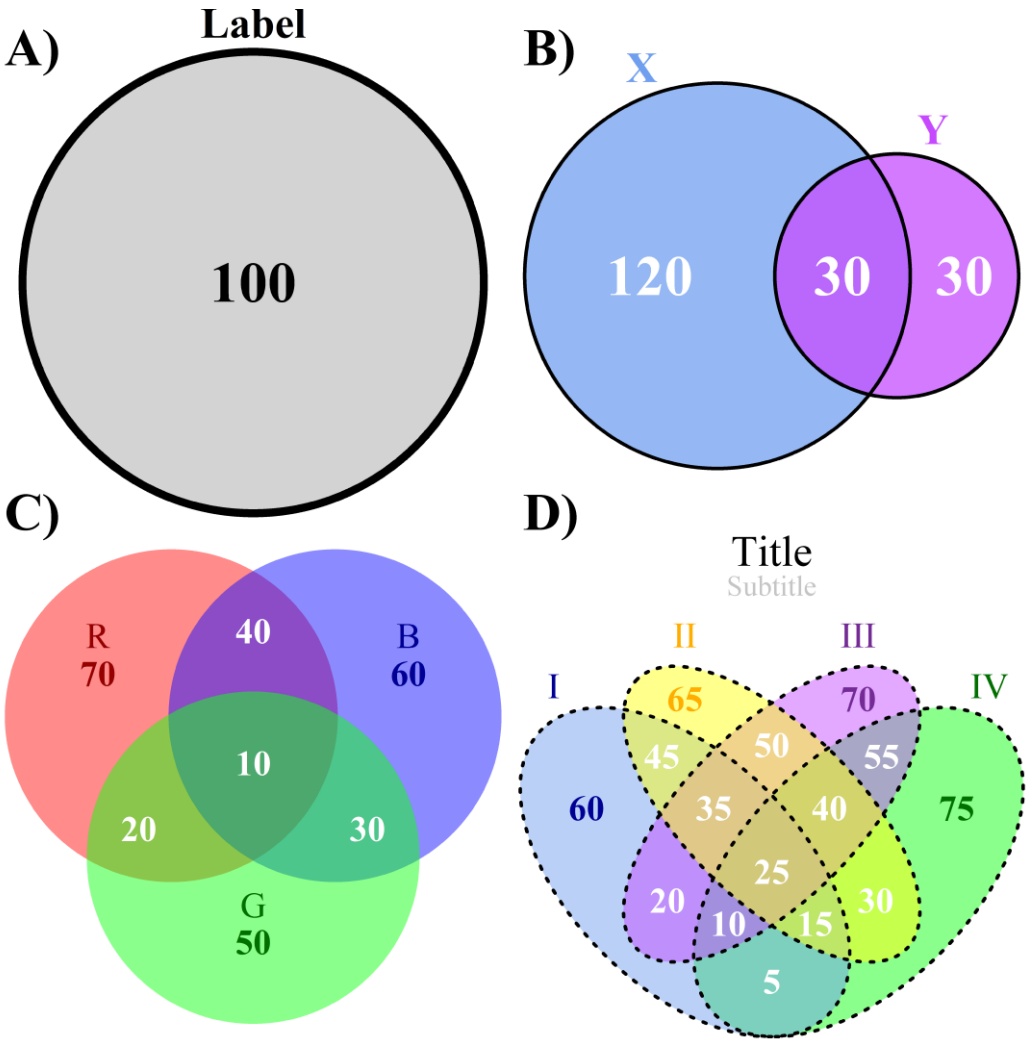

Figure 4: The four types of Venn diagrams drawn by the VennDiagram package[44]

2.3.3 VennPainter (InteractiVenn)

Set comparisons permeate a large number of data analysis workflows, in particular, workflows in biological sciences. Venn diagrams are frequently employed for such analysis but current tools are limited. [42] InteractiVenn is a web-based tool that aims to solve this problem. One key distinctness is that when established Venn tools are limited to four-sets, groups, InteractiVenn can visualize up to six-sets. Nevertheless, similar limitations mentioned above such as interpretation and limited group number is still valid for InteractiVenn as well.

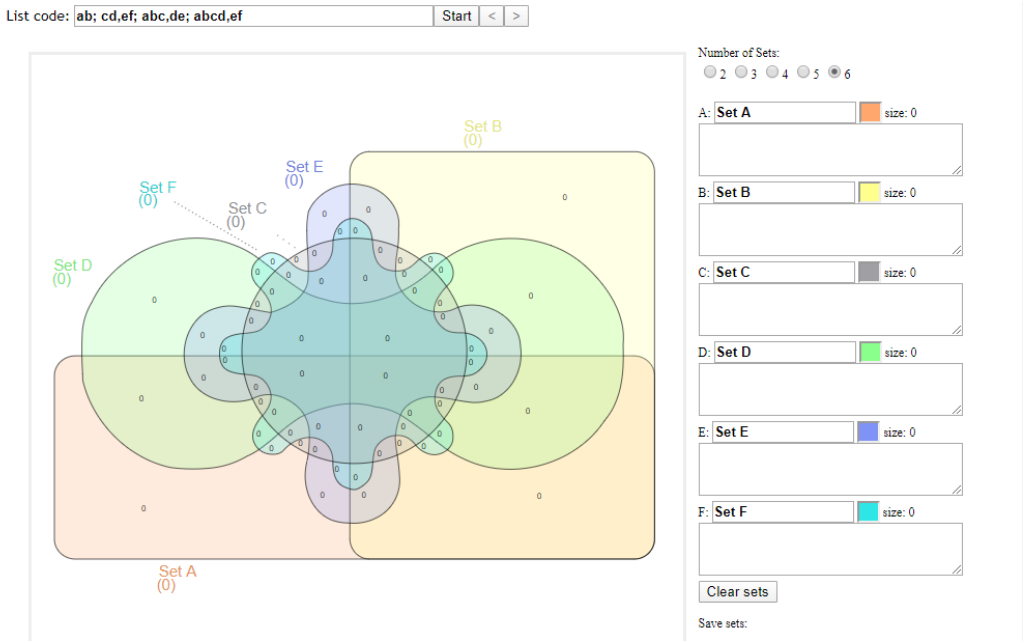

Figure 5:Opening Page of InteractiVenn [42]

2.3.4 GeneVenn

GeneVenn is a web-based Venn Diagram creating tool. [45] Although there is not any publication related to this tool, from the outdated website and interface we can extract some information. This program extends its functionality with couple of features which many online tools has not got. GeneVenn can list the elements in each region but it can do only one region, it cannot show whole cluster at once. Also, users can add gene lists to “a text area” as a set of strings and also upload the system ASCII files with a guideline. Uploaded file needs to separate genes with “any white space like space, tab, or line break, and comma”. If user upload a file and uses text area for cluster, program counts both of them as a one single cluster. In spite of that, this tool can compare only three clusters, gene group. Program does not allow saving created diagram in any way, user needs to come up with their own idea which is taking a screenshot of the window. Additionally, since it is released in 2006, there hasn’t been any changes and because of that,

programs user interface (UI) is outdated. There hasn't been any citation style specified in the website.

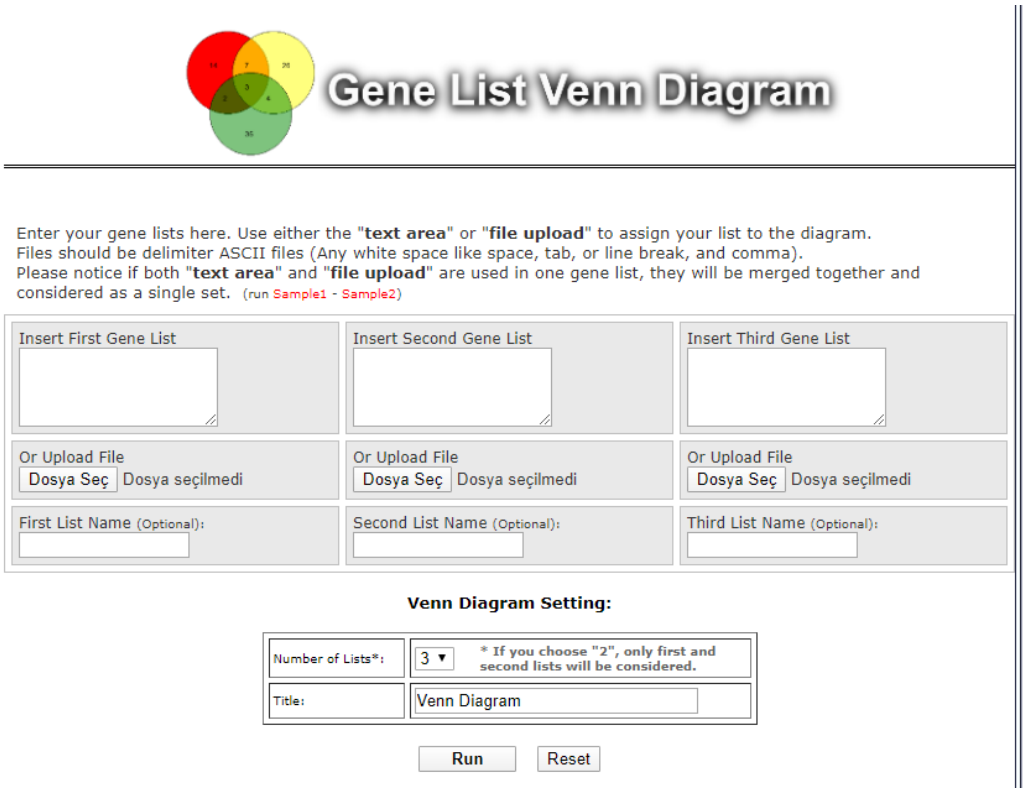

Figure 6: GeneVenn Website Opening Page[45]

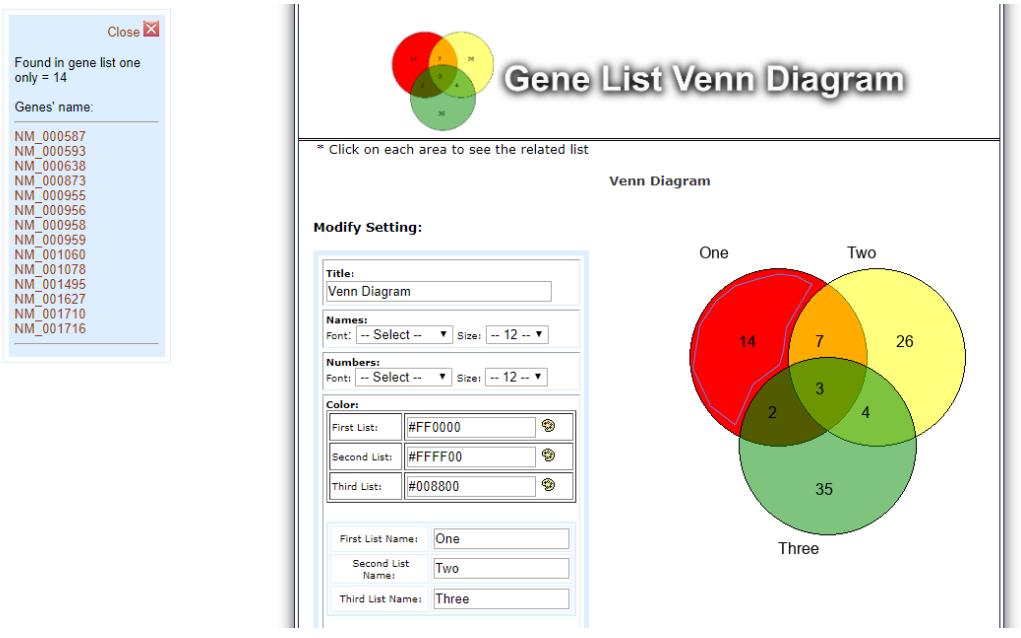

Figure 7: Main difference from other tools, listing elements in specific region[45]

2.3.5 BioVenn

BioVenn is a web application for the comparison and visualization of biological lists using area-proportional Venn diagrams. [46] It has couple of customization with basic drawing and styling. Although it has been published and developed in 2008, BioVenn differs with offerings such as font type and size selection, exporting diagrams as embedded SVG, embedded PNG, SVG Only, PNG Only options. In spite of that, BioVenn can compare and lists maxium three sets of lists. Additionally, user interface seems to be outdated and it can show indivual elements that are in the proportional areas.

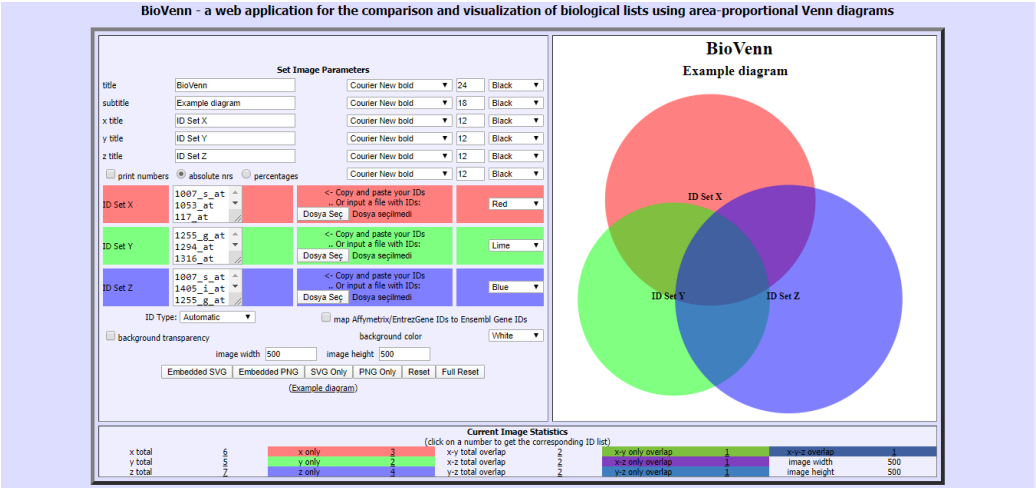

Figure 8: BioVenn's user interface and built-in example drawing [47]

2.3.6 VennMaster

“VennMaster is a tool for drawing area proportional Venn/Euler-diagrams. It supports several input formats from simple tab seperated data to gene lists from GoMiner. VennMaster will generate area proportional Venn diagrams for multiple sets. Since exact solutions seldomly exist for more than 3 sets, it will approximate a correct solution. VennMaster is great for assessing overlapping data at a glance, as well as for in depth analysis.” [48]–[50]

VennMaster is a Java Application which can run any operating system. The software requires the Java runtime environment >=1.5.0 (JRE 5.0). One pros to application is that it is integrated with GOMiner in the context of the Gene Ontology database. Even though it is cross-platform application for the reason that being developed as a Java Application; since it is built in 2005 and last updated next year; whereas, with not supported libraries, all new dependencies, required installation time and dealing with possible errors, out-of-date user interface makes VennMaster is not a suitable candidate even for drawing Venn Diagrams.

2.3.7 jVenn

jvenn is a basically JavaScript library. It processes lists and produces Venn diagrams. It handles up to six input lists and presents results using classical or Edwards-Venn layouts. User interactions can be controlled and customized. Finally, jvenn can easily be embeded in a web page, allowing to have dynamic Venn diagrams. [51] jVenn developed as a jquery plug-in. [52]

While we asses bioinformatics tools, in terms of efficiency and usability, jVenn stood out from other tools that does the similar tasks. This conclusion draw from following key features; handles up to 6 classes venn diagram, allows to provide the data from 3 different ways (lists/intersection counts/count lists), control the click callback function, provides statistic charts based on input data, search for elements, exports the venn diagram to PNG and SVG, exports lists to CSV. In spite of that jVenn does not perform network visualization related tasks and therefore it lacks one of the crucial tasks, this thesis wants to achieve.

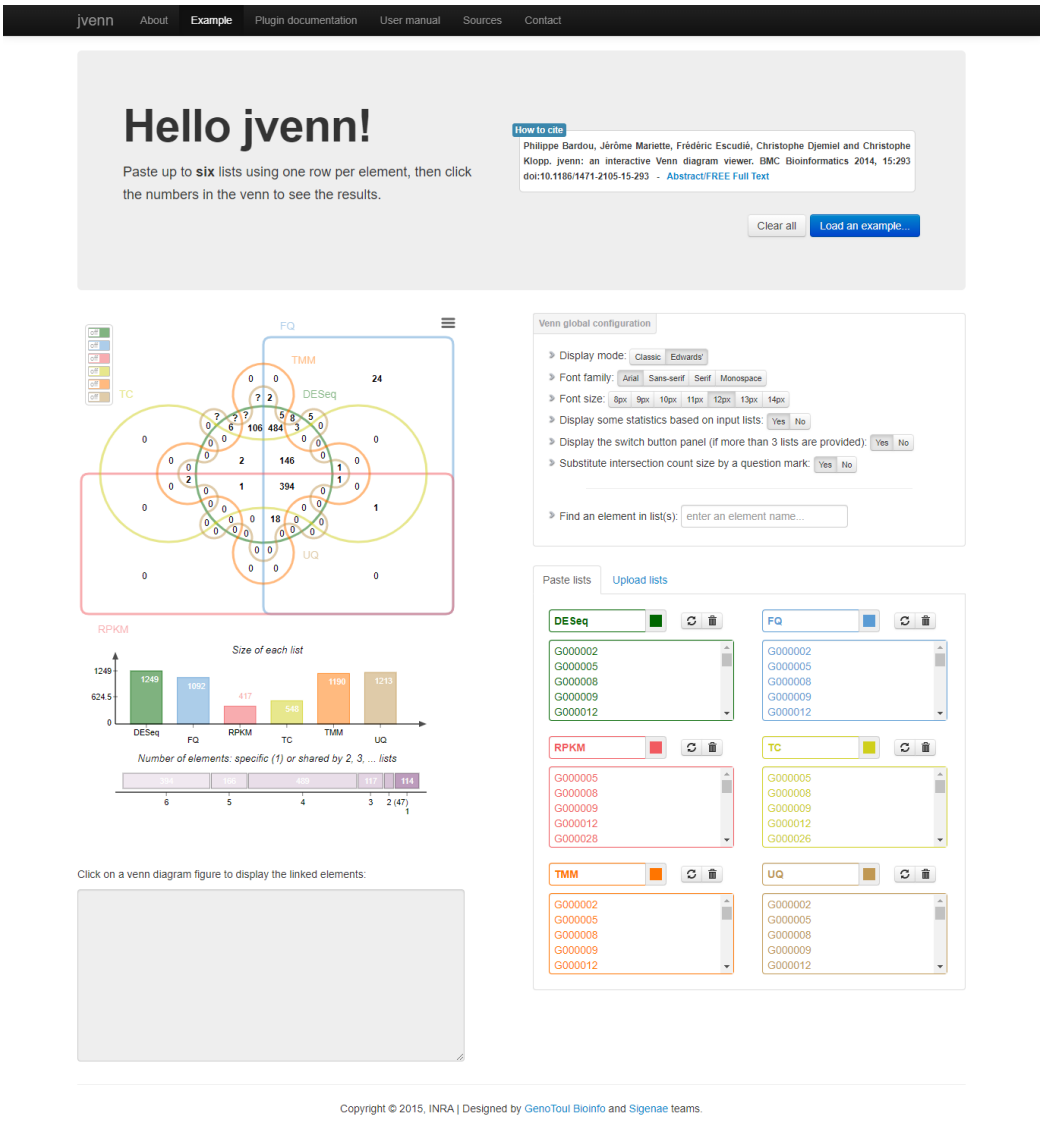

Figure 9: jVenn web application and built-in example with Edwards' Diagram [51]

2.3.8 Tom Sawyer Visualization Tool

Tom Sawyer Software is a tool for visualizing maps and statistics and for analyzing social networks. In order to recognize development opportunities and challenges, companies use these tools to consider relationships, patterns, and behaviors to complex data sets. Tom Sawyer Software is used by banking, accounting, resources, engineering design, security and technology, health and

life sciences, networking, industrial technology, and manufacturing organizations. [53]

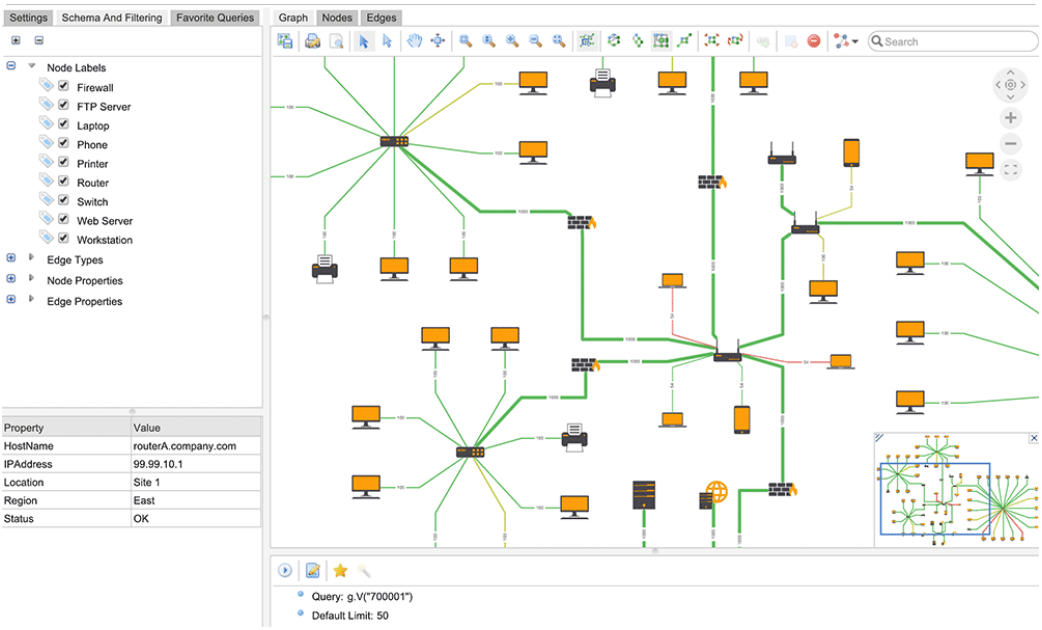

Figure 10: Tom Sawyer AWS Neptune[53]

Tom Sawyer Software is a commercial product that does the analysis. Users can use the tool for 60-day evaluation but due to product complexity, it requires some expertise while using it. There is no fixed price for the tool as well, in order to access full software, you need to contact the product sales team as well.

2.4 Package Based Visualization Tools

Understanding interactions between proteins, genes and enzymes are the key for understanding humans, our reactions to diseases and drugs. Because human body is a system that all of it parts integrated with another. Therefore, we cannot simply explain a mechanism with one aspect of it. Venn diagrams can help us only a statistical way. In this chapter we will look at some of the bioinformatics tools that have similar aims like BioNetVis. You will see some of the tools are really complex in require interdisciplinary knowledge, some of them are basic tools. We will continue explain tools like Venn Diagrams for continuity, later will take a peak to more complicated ones. Because of that, Venn Diagrams itself are not

efficient way to understand protein-protein interactions, similarities between diseases, and genes. This thesis aims to visualize similarity between diseases, protein-protein interactions in order to illuminate path for both treatments of diseases and personalized medicine, drug discoveries.

2.4.1 VennPlex

VennPlex, a program that illustrates the often diverse numerical interactions among multiple, high-complexity datasets, using up to four data sets. VennPlex includes versatile output features, where grouped data points in specific regions can be easily exported into a spreadsheet. This program is able to facilitate the analysis of two to four gene sets and their corresponding expression values in a user-friendly manner. [54] Although VennPlex seems to be a convenient choice, VennPlex has many limitations. These are; it is a package based program and works only computers which has Windows operating system. Program requires expression values of genes as an input in order to work, thus makes it impossible to work for proteins, genes without their expression values. (CHECK THIS ONE) As a result of this requirement, data that collected gets bigger and unintuitive to read. Moreover, it can handle maximum four datasets and this is unefficient and out-of-date as you have seen chapter 2.3, there are many state-of-the-art tools such as jVenn, to work with. Since it does not support cross platforms, do not do any visualization other than Venn diagrams, VennPlex becomes inadequate to work with as well.

2.4.2 Vennture

VENNTURE, a program that facilitates visualization of up to six datasets and it has ability to export diagrams as spreadsheets. It is capable of a highly complex parallel paradigm, i.e. comparison of multiple G protein-coupled receptor drug dose phosphoproteomic data, in multiple cellular physiological contexts. [55] There are many downsides vennture in terms of usage. Program requires Excel files with corresponding column for expression values and since it works only operating systems on Windows. Therefore, even to get a simple Venn Diagram,

let alone network visualization, these limitations itself makes Vennture less popular choice while drawing venn diagrams manually.

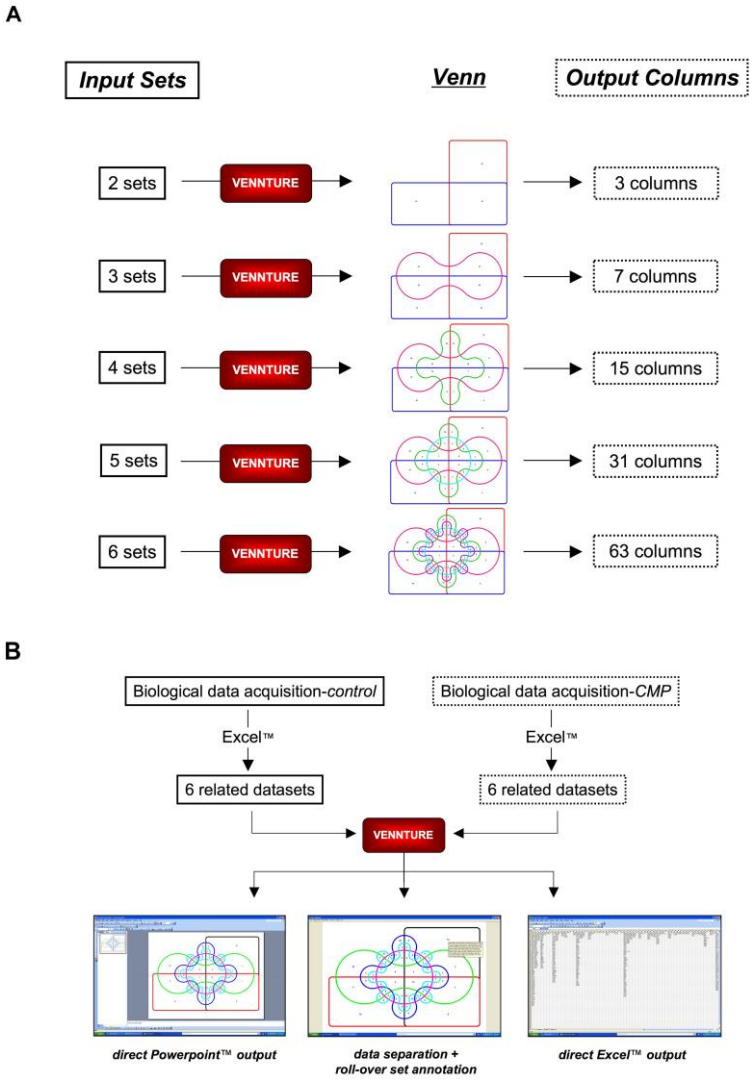

Figure 11: Multiple set VENNTURE data input. Figure taken directly from corresponded article. [55]

2.4.3 Cytoscape

Cytoscape is an open-source software platform for visualizing molecular interaction networks and biological pathways and integrating these networks with annotations, gene expression profiles, and other state data. [56] Although it has been originally intended for biology research, now Cytoscape is a global platform for complex network analysis and visualization. In terms of the working environment, there are a couple of versions of Cytoscape such as Cytoscape

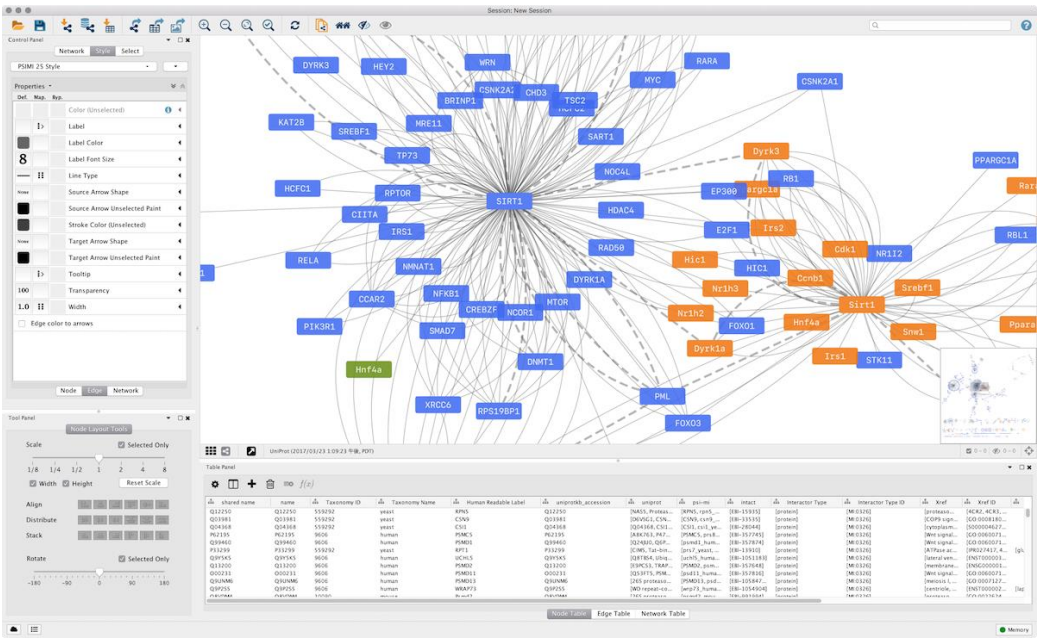

Figure 12: Cytoscape 3.5 Desktop Application Overview [56]

Cytoscape Desktop 3.5 software, web-based version Cytoscape.js which is the successor of Cytoscape Web has similar applications to Cytoscape Desktop version, hence we won't be mention for a web-based application.

Although Cytoscape started for biological research purposes, with the help of being an open-source platform and giving access to the developers to add features with plugins, they broaden its abilities. There are many advantages and disadvantages to this which also will be discussed in section 3.

2.4.4 Gephi

Gephi is an network visualization and analysis tool developed on Java programming language with an open source availability. [57] The tool started to developed an university environment, like BioNetVis, by the students of the University of Technology of Compiègne (UTC). Among its research purposes, Gephi also can be used for other visualization applications such as journalism, political research, history and such.

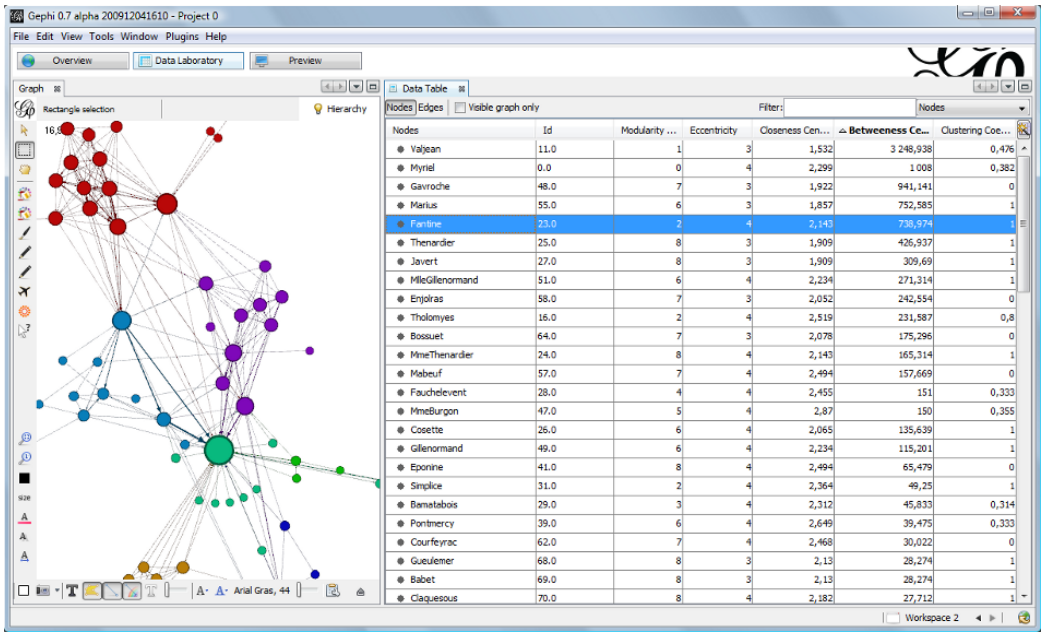

Figure 13: Gephi Package Program A Visualization Screen [57]

2.4.5 NetworkX

NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks. It is mainly focused on the scaling specialty which can be used in real-world problems within graphs in excess of 10 million nodes and 100 million edges. Therefore it is suitable for handling large datasets. [58]

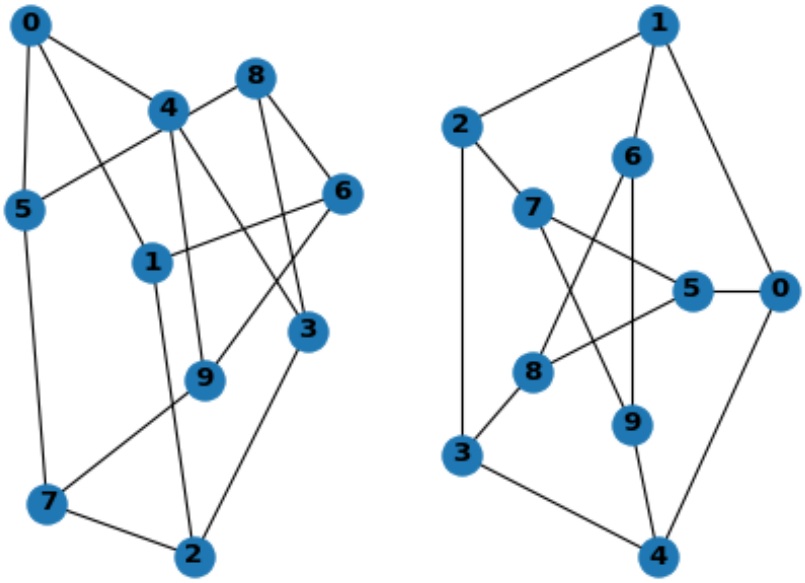

Figure 14: A Graph created by using NetworkX [58]

2.4.6 NodeXL

Node XL is a plug-in developed in .NET language for the analysis tool for various Microsoft Excel versions from 2007 to 2016 that provides network visualization, social network analysis, content analysis, task automation, and such tasks. In 2015 the company divided the tool for free and non-free versions for several reasons. These two models are NodeXL Basic and NodeXL Pro. NodeXL Basic is available freely and openly to all. It is positioned as a browser for files created with NodeXL Pro which offers advanced features for professional social network and content analysis. NodeXL is basically a set of previously built class libraries using a custom Windows Presentation Foundation control. [59]

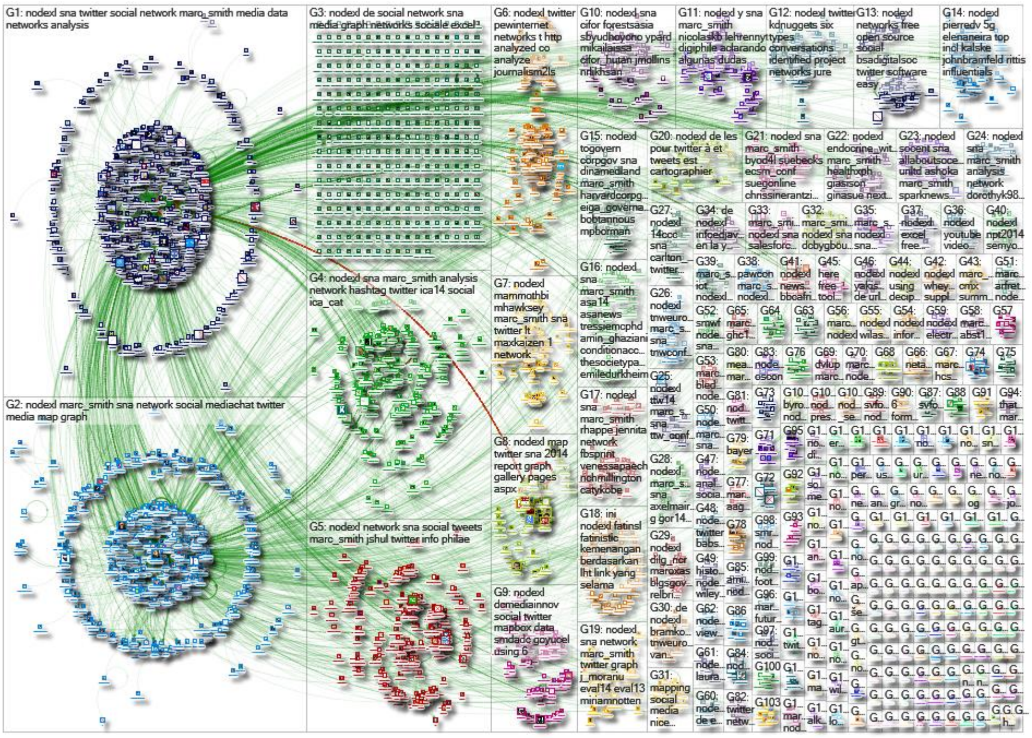

Figure 15: Different set of visualization graphs created by using NodeXL [59]

2.4.7 Graphviz

Graphviz is open source graph visualization software. Graph visualization is a way of representing structural information as diagrams of abstract graphs and networks. It has important applications in networking, bioinformatics, software engineering, database and web design, machine learning, and in visual interfaces for other technical domains. The Graphviz layout programs take descriptions of graphs in a simple text language, and make diagrams in useful formats, such as images and SVG for web pages; PDF or Postscript for inclusion in other documents; or display in an interactive graph browser. Graphviz has many useful features for concrete diagrams, such as options for colors, fonts, tabular node layouts, line styles, hyperlinks, and custom shapes. [60]

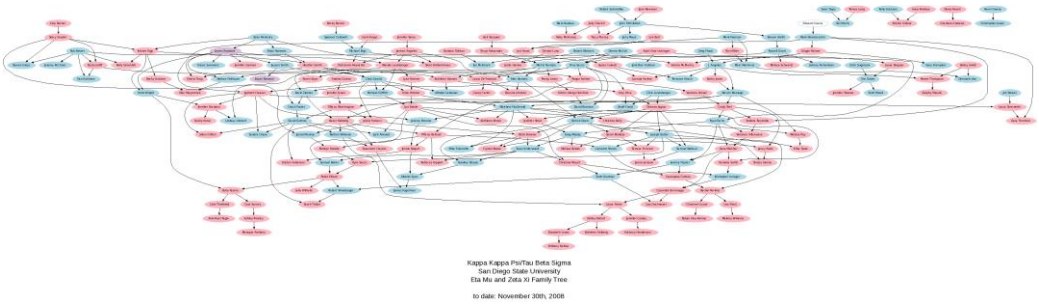

Figure 16: A Family Tree graph visualization created with Graphviz [60]

2.4.8 upsetR

Visualizing the sets and their intersections is frequent challenge for academics and researchers whom works on the biological and biomedical data. Even there are some technics for visualization, often these technics fallen short in terms of complexity and usability. [61] These approaches often has shortcomings such as inability work with large number of sets, interpretation of the data from generated visuals are often hard and open to discussion. UpsetR is a package based program developed for R programming language and software. [62] It is inspired from "Upset" graph technique created by Lex et al. [63], [64]. Upset technique's layout is focused on showing intersections with matrix-based approach. The package based programs has supports three input formats; a table in which the rows represent elements and columns include set assignments and additional attributes; sets of elements names; and an expression describing the size of the set intersections as introduced by the venneuler package. [62], [65]. Some may argue, the disadvantage of upsetR is that loading and reading data is inconvenient and understanding visualization is more complex while it points out Venn and Euler diagram's shortcomings in terms of interpretation. Figures below shows a

visualization made with upsetR on R program.

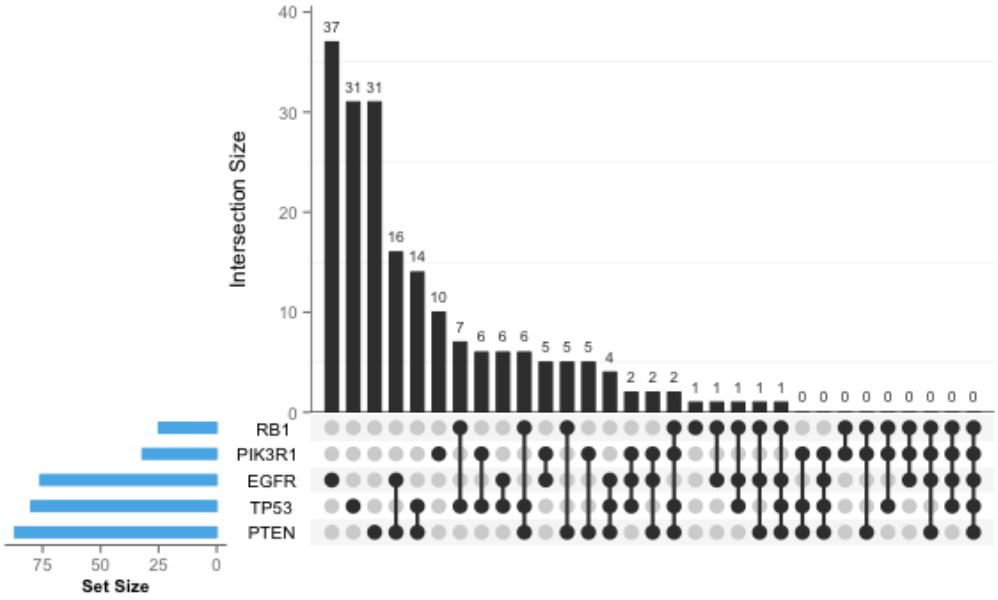

Figure 17: A visualization made with upsetR. Figure taken from article's supplementary GitHub page. [62], [66]

Chapter 3

Implementation and Methods

In this section you will find our approach to the projects actual development part, which is basically coding. This requires simple introduction to the materials that we have been used. Thus, firstly we will briefly give an explanation to the architecture of BioNetVis. It has main two parts, for creating the network from the data back-end section and for interaction with user and visualization front-end part. Above all, later in this chapter you will find methods and objects that have been created can be seen in the regarding sections, and you can find the written codes in the GitHub repository of the project. [67]

3.1 Architecture of BioNetVis

BioNetVis has two main parts. Firstly, a network needed to be created in order to visualize it. Secondly, in order to visualize there has to be an interaction between user and proposed tool. For this reason, other part is creating an interactive user interface so that user can give inputs to visualize. In below figure 16, you will see simplified version of a flowchart back-end parts and at the figure 17 shows flowchart for the front-end section.

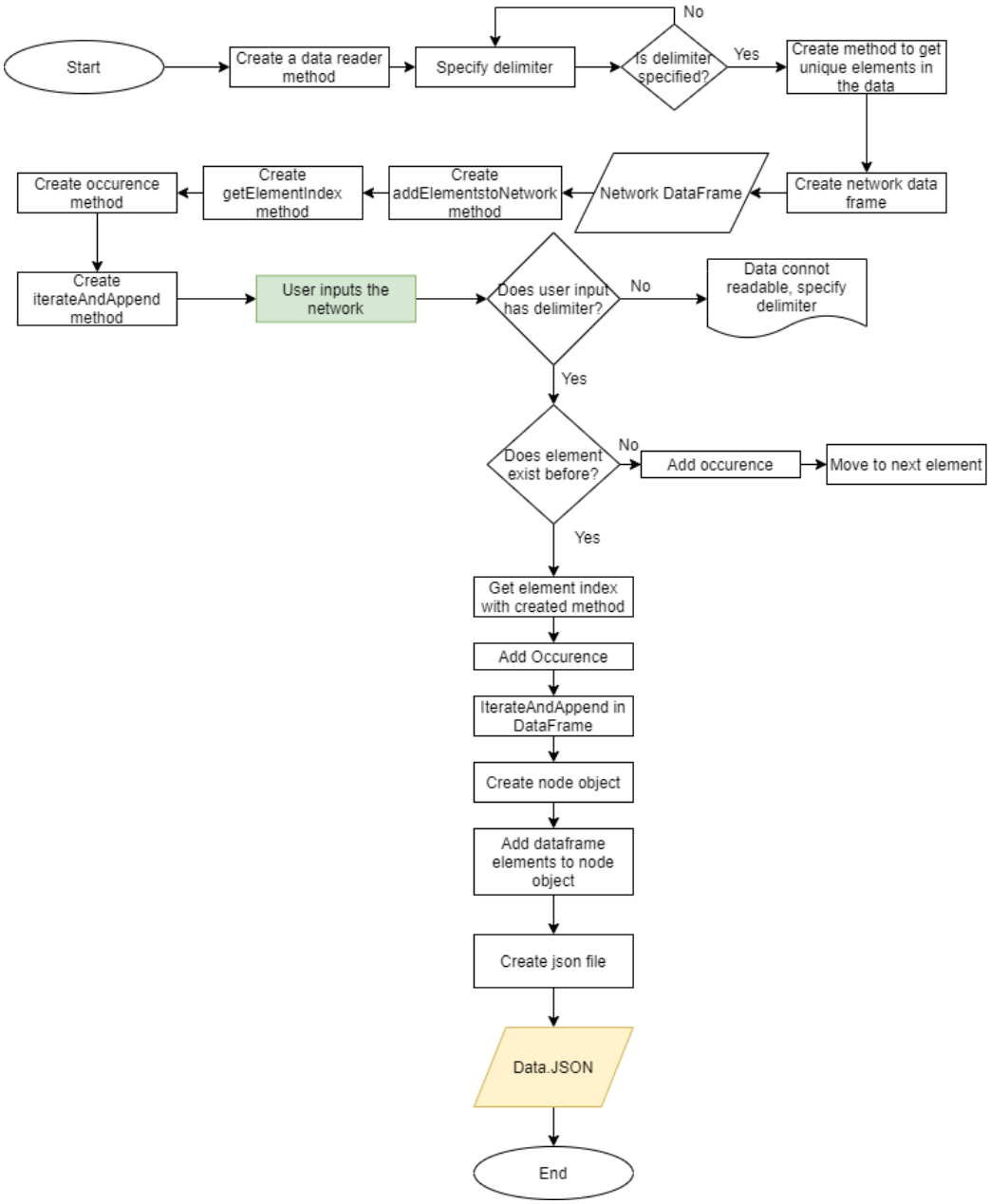

Figure 18: Simplified version of back-end part of the thesis work

While we start coding, first task needed to be done is creating methods for generalizing. After initial release, there can be following updates so making codes open ended will enables performance for future developments. At figure 16, created methods are not taken because of space issues in the figure and for thesis. Firstly, we created a method for reading files, especially .csv files. While reading these files, it is required to specify delimiter of the object that will feed to the

program. Additionally, the input file's header also needed to be considered. Each element in the data has to be unique or needed to indicate it is repeated data. For this purposes evaluation of unique element method is created. If read data is unique it will have iterated and append to the also created network dataframe with its index. If read element is occurred second time it is specified in the another occurrence dataframe. For the first version of the BioNetVis user has to input its network file manually. In this case our input file is more than 62,000 lines and each line has two elements with a "pp" delimiter which indicated both elements are neighbors and have connections. To get this information a node object is created and for visualization purposes in the front end, each object has to have "id, label, size, color and x, y coordinates" information. Since color and coordinates are related to visualization, to save up from the memory and efficiency purposes these 3 elements of node object will be added in the front end section, with JavaScript language and sigma.js framework, also will be mentioned in the materials section. Creating this network requires computing power and needed to be done carefully, so sparse matrix is used to create connections between nodes and adding id, label and size information. After iterating and creating each object all of the information "dumped" to a JSON file which front – end section will be use.

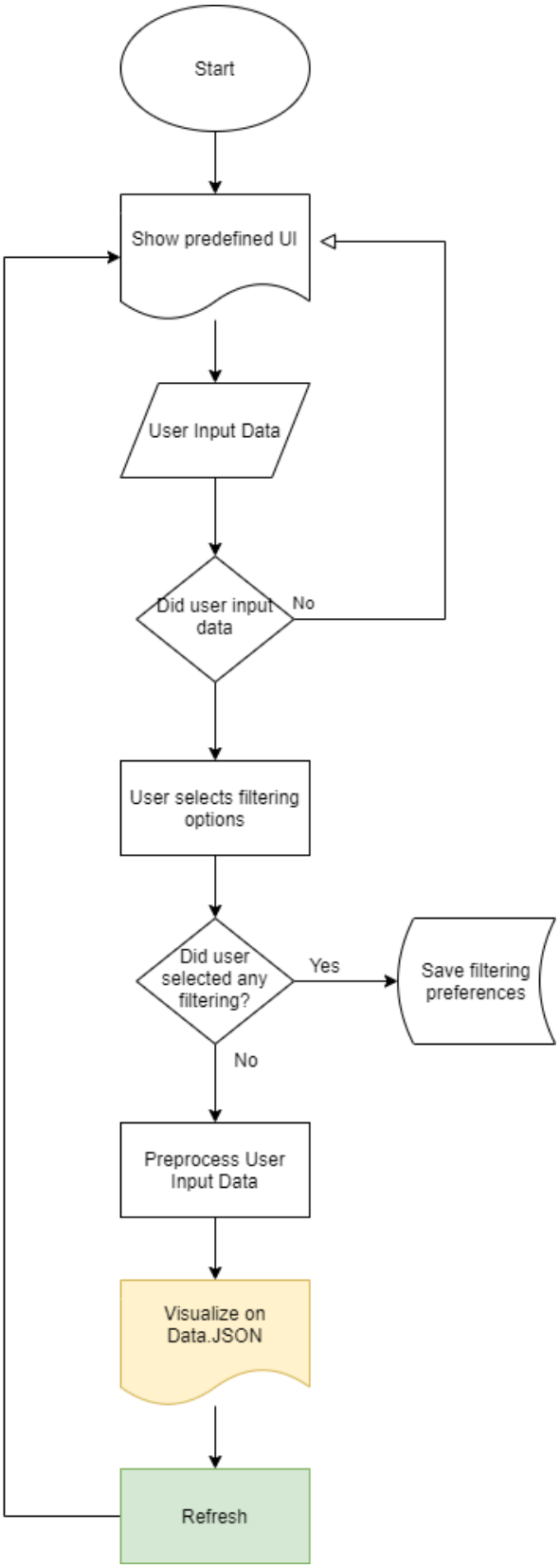

Figure 19: Front-End part of the thesis work

For front end section a simple, understandable user interface needed to be created. Several instructions for how to use BioNetVis is also added for making usability. A predefined user interface is created and when user will visit the project's website, it will be loaded. Predefined User Interface is combine of a header and text block, a filtering area, six text area for user's input with modern color picker boxes, a visualized protein-protein interaction network with randomly generated coordinates, a toggle button to arrange coordinates of the nodes, exporting button, information panel for nodes and an information block for to properly cite thesis work. User can put inputs which is proteins and can select the color for visualization. Additionally, user can select filtering option before or after visualization. Based on user's preferences the tool preprocess and visualize on the network which we get the information from the back-end section. In result of that user can export generated network or can refresh the page for a new query.

3.2 Material

There are many software development process methods for building a project. Some of them are Agile, Waterfall, V-Shaped, Spiral, Evolutionary Prototyping, Incremental, Random Modelling and such. These methods are invented in order to fulfill development team's needs while considering many aspects of developing, such as, resources, design, architecture, feedbacks, time limit and many more.

In this thesis work we relied on waterfall method since from the beginning of its goal of the thesis were specified, there were information about what needs to be done, time limits and feedbacks are having to be related to goal of the project.

In this chapter three, we are going to investigate development aspect of this study. What language/s have been used, which editor chosen, what are the frameworks and which libraries in use in this program. Also, in addition to that, we are going to explain why these specific choses have been made.

All of the work that will be described below is executed in Windows 10 Pro Environment with additional programs and installments, can be repeated with same exact conditions.

3.2.1 Languages

Programming languages are formal languages that are includes a collection of instructions that produces different output forms. Computer programs uses programming languages in order to implement algorithms and execute tasks. There are numerous programing languages such as Java, Python, Ruby, C, C++, C#, Swift, Go and such. Complexity of the programming languages differs interpretation of human readability and computer readability. Complexity is a different issue which we will not be covering this work but keep in mind, while we refer complexity in programming languages, it implies readability to human and ability to perform complex task without many lines of code blocks.

3.2.1.1 Python

Python is an interpreted language with expressive syntax that some have compared to executable pseudocode. [68], [69] Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. The language structure and object-oriented methodology are aimed at enabling programmers to write simple, functional code for small and large scale projects. [70]

There are many specialties and features that Python language has. These are;

An open source license that allows use, sell or distribute Python based application without extra efforts for permissions

It can work all sorts of environment, that means while writing an application user does not have to worry about limited portability and vendor lock-in.

Language’s simple yet complex syntax allows you to build programs for different needs whether fully object oriented based or local necessities.

Powerful interpreters can speed up the developing process with real time code development and instant experimentations, therefore it rules out time consuming step of test development process.

Python can be taught, means added, more tasks with user's own complied codes allows it can do anything as much as hardware reaches.

The language can be easily integrated with prevail programs thus easy to use with older applications as well.

It can interact with wide range of different programs and languages on the operating system which allows flexibility and advantage in terms of using software skills that may have been acquired before.

Python has large number of library modules (that comes with installation of language and also can be installed via additional downloads) allows developers to built complex programs like from solving high level mathematical equations, image processing tasks, building machine learning models, calculating satellites orbits to genome sequencing.

Daily growing Python Community is helpful in many aspects, like learning the language, quickly tackling code-problem solving issues and more.

All of the Python modules are can be found at pypi.org with easy to installment packages. [69], [71].

Without getting into deep subjects like formatting, operations, syntax and more, it is clear that Python is one of beneficial language for all kinds of projects from different backgrounds. Besides the large abilities, in thesis work, Python is used for back-end part of the project and that is the backbone of the project. This language has been chosen numerous reasons but main ones are; long life of the language for feature projects as well, understandable code for everyone and last but not least, my familiarity with the language itself.

In this thesis work Language's Python 3.7.0 version (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32 is used. Used complier and libraries will be mentioned in respect of their subjects.

3.2.1.2 Javascript

JavaScript is a programming language with complex features such as compiling instantly, multi-paradigm specialties and high-level complexity. The programming language has syntax that mostly works with curly brackets, object orientation, dynamic typing and first-class functions. JavaScript, can be referred as JS from now on, follows a set of rules for standardization and these set of rules are ECMAScript specifications. [72], [73]

In many aspects we can say JS is the Web's programming language. The immense amount of modern websites and also all modern web browsers on any devices uses JavaScript. In result of that usage for all modern devices, desktops, tablets, smartphones, game consoles make JS one of the pervasive programming language of the software developments. The language often mentioned and related with other World Wide Web (WWW) display languages. A web developer must have information for three languages and these are; HTML for specifying constituents of web pages, CSS for appearance of the web pages and JavaScript for specifying behavior of the web pages. [74]

JavaScript language has been chosen for this thesis work for following reasons. It's interactive specialties makes it a must for visualization projects. Every web page requires interactions with users and to keep user's attentions animations. The language is also compact and fast and thus JavaScript is a must.

3.2.1.3 HTML/CSS

HTML is an acronym for Hyper Text Markup Language. HTML documents, the foundation of all content appearing on the World Wide Web (WWW), consist of two essential parts: information content and a set of instructions that tells your computer how to display that content. [75]

HTML documents from a web server or local storage unit is read by web browsers and rendered and showed as multimedia web pages. This documents tells browsers the appearance of a web page semantically and adds information cues for the structure of the web page.

Building blocks of HTML pages are the HTML elements. In order to denote elements by tags, the language uses angle brackets. HTML elements can be in many forms such as images, texts, interactive forms, pointers, headings, paragraphs, lists and more. For example, in order to indicate an heading to a paragraph users has to use <h> Title of Heading 1 </h> and Some paragraph text tags. With this tags, browsers can interpret the content inside the brackets and display them on the webpage.

It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. [76]

Cascading Style Sheets is a style sheet language used to characterize the text written in a markup script, such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. [74]

CSS is designed to enable the highlighting different representation for content, inclusive of layout, colors, and fonts as well. [77] This highlighting separates the content and improves it via accessibility, gives more flexibility and management in the specification of presentation characteristics. With combining all of the relevant codes in a separate .css file allows multiple web pages the sharing same formatting by specifying relevant parts in the file. Hence it also reduces complexity as well as repetition in the code blocks. [78]

In order to upload thesis work that has been done for the interaction between researchers and BioNetVis, for showing the content of the website HTML is a requirement. For appearance and quick referencing all the elements in the HTML and JS, CSS language is a requirement as well. In the source files of the thesis work you will find respective files for the codes that have been written for the project.

3.2.2 Integrated Development Environments (IDEs)

Source code editor is basically a program specifically designed to edit computer languages codes. It can be a stand-alone desktop application or it may have built on top of integrated development environment (IDE), even can be embedded into

a web browser. Source code editors are essential for programmers since it is the actual workplace to do so.

These source code editors are especially designed for making faster the typing or simplifying the code with features like brace matching, indentation, syntax highlighting and autocomplete functionalities. In addition to that, source code editors also maintain adequate practices to run a compiler, interpreter, debugger or working with relevant 3rd party programs for software development process. Even though there are many text editors such as Notepad to edit code, if they cannot improve coding process with features like automating, easily editing multiple line of codes, they cannot consider as source-code editors. Instead they are simply text editors. [79]

There many code editors you can find along with many specialties in wide range of usage such as free to use, pay to use or free using but required license while selling, monetizing your code. In this chapter you will find the IDEs and source code editors that has been used in thesis for different purposes. Editors will be explained briefly and reasoning for choosing specific editor also will be given as well.

3.2.2.1 PyCharm

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. The program developed and released by a Czech technology company called JetBrains. [80] "It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as Data Science with Anaconda." [81]

PyCharm works with macOS, Linux and Windows operating systems that makes it a cross-platform IDE. "The Community Edition is released under the Apache License and there is also Professional Edition with extra features – released under a proprietary license." [82], [83]

Some of the features of PyCharm are;

- Writing assistance and analyzing via code completion, suggestions, highlighting syntaxes, errors and quick corrections
- Easy navigation between projects and codes, specially designed project and file structure views, quick jump to methods, classes, objects
- Python refactoring; renaming, extracting methods, introducing variables, constants
- Integrated web frameworks, such as Flask, Django and web2py (requires professional edition)
- Integrated Debugger and line by line code coverage, Unit Testing
- Google App Engine Python development (requires professional editon)
- Version control, UI for Git, Mercurial, CVS with change lists and merge
- Scientific tool supports such as: matplotlib, numpy and scipy (requires professional edition) [84], [85]

Some of the features mentioned above can be found at other IDE's as well. Nevertheless, ability to work with web frameworks and scientific tool support is a must for thesis work. In order to integrate front-end and back-end codes, to be able to make code work smoothly, finding and fixing errors quickly web framework feature is important. Also while pushing codes online in time perspective it will be helpful. Moreover, while working with Human PPI Data, there are more than 62.000 lines of information. Creating a network with this amount of data is overwhelming. Keep that in mind, there are also additional clusters, data-sets which in this case user will provide, can go up to thousands of proteins, genes. That extreme amount of data needs to be handled carefully, every element in the data should be searchable in the PPI Network. This utmost data processing it requires powerful IDE and decisive scientific tools. In light of these factors mentioned above, PyCharm is selected for default source code editor in the thesis work.

Used Pycharm version is Professional Edition, licensed with educational purposes. License and used version is specified both written and showed below. PyCharm 2019.3.2 (Professional Edition), Build #PY-193.6015.41, built on

January 21, 2020, Licensed to Umit Bulut, Subscription is active until February 3, 2021, For educational use only. Runtime version: 11.0.5+10-b520.30 amd64 VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o, Windows 10 10.0, GC: ParNew, ConcurrentMarkSweep, Memory: 978M, Cores: 4.

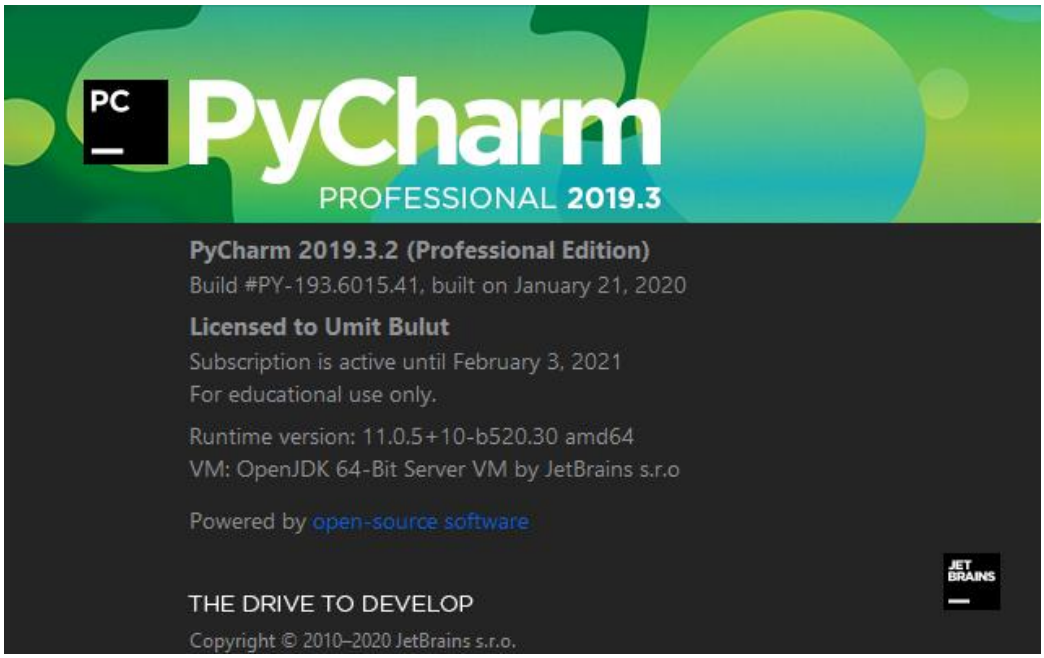

Figure 20: Pycharm IDE Version and License information [80]

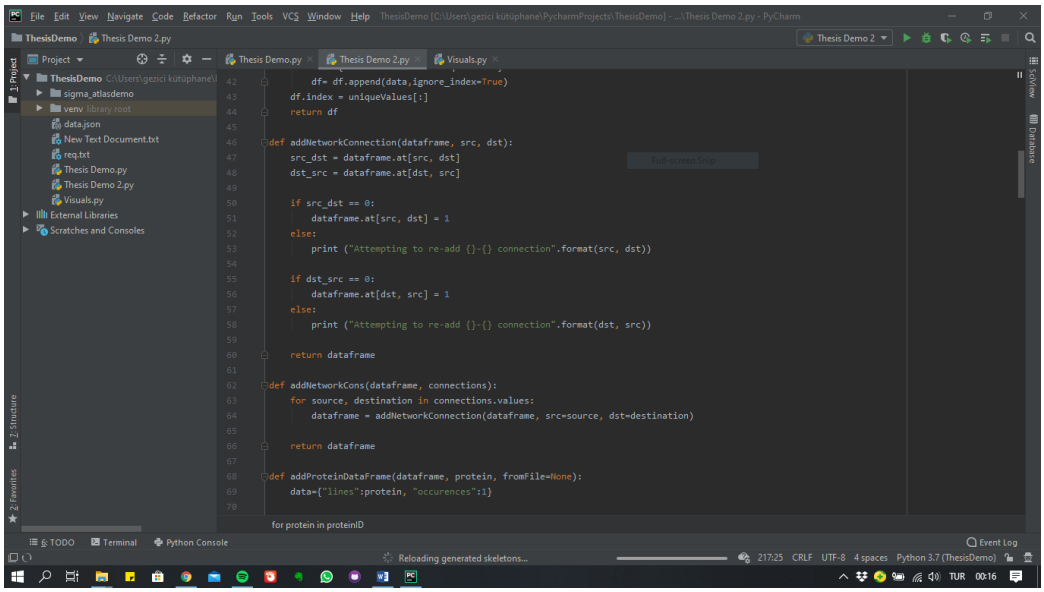

Figure 21: Pycharm working environment while working on back-end development [80]

3.2.2.2 Codepen

Codepen is an online code editor for web development with several features that allows users to code, test and showcase their work. [86] It allows to work with HTML, CSS and JavaScript code snippets, and call these code blocks as “pens”. The changes in the code snippets can be seen and test in real time. Indenting, pointing opening and closing methods, brackets are helpful features as well. It also has a social function where users can share their work and communicate each other through comments. Codepen is developed by two front-end developers and a front-end designer; Alex Vazquez and Tim Sabat and Chris Coyier in year of 2012. [87]

In web development part of the project, to be able to interact with the visualization part via JavaScript, seeing changes in real time without losing time and lastly, creating, giving a shape to a webpage on a webpage itself was helpful while developing this thesis work, BioNetVis. For that reasons Codepen is also has been used as an online code editor.

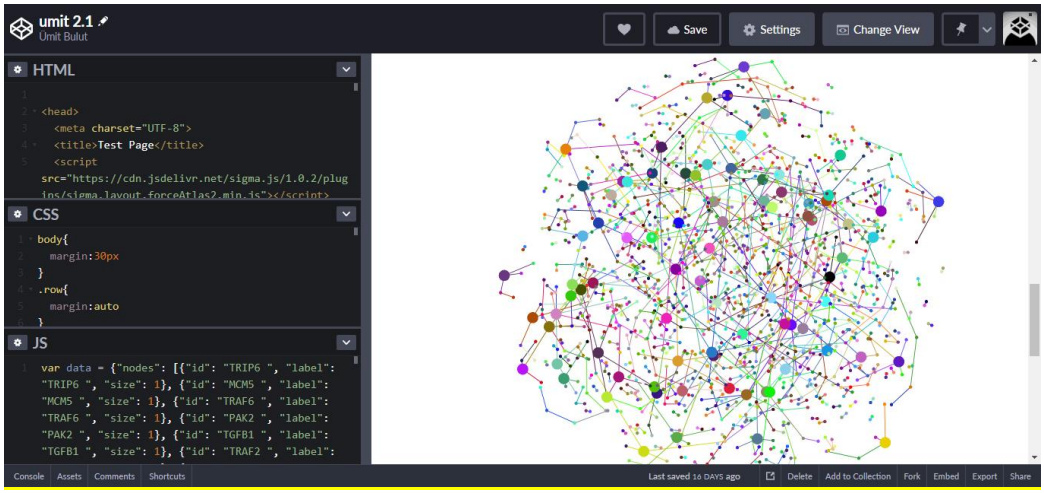

Figure 22: Codepen working environment while working interactions between nodes and UI[86]

3.2.3 Frameworks

Frameworks are the services that allows developers to integrate their works, their codes into a system. In this section we are going to look at frameworks that has

been used specially for this thesis. These frameworks are a necessity for the front end aspect of the BioNetVis. Like other materials explained above, reasons behind chosing exact frameworks will be given after introducing their features. Relevant topics will be followed as, Flask, jinja, Bootstrap and Sigmajs.

3.2.3.1 Flask

Flask is a micro Web Server Gateway Interface (WSGI) web application framework written in Python language. [88] The reasoning behind "micro framework", Flask does not require any additional tools or libraries in order to work. It is designed to start web related projects quickly and easily with being able to scale up to complex applications as well. In the initial release, Flask started as basic wrapper tool around Wekzeug and Jinja, currently become of the most popular Python web application framework. Platforms like Linkedin and Pinterest are use Flask as well. [88], [89]

Flask has not have any database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

On the other hand, Flask can have a feature to support third party extension. In example this feature allows extension itself implemented while Flask is coded since Flask is simple start and run, micro framework. There are extensions for different purposes such as validating forms, open authentication technologies, upload handling and object-relational mappers are can be used with Flask. Since these extensions answers needs of developers, they are updated far more frequently than the core of Flask framework. [88]–[91]

Figure 23: Flask framework installation and documentation page[89]

Flask has been chosen its' simplicity and lightweight features. It works as a pipeline for the Python codes to work with front end codes. In addition to that, it is not complicated for stating point while comparing with frameworks like Django.

3.2.3.2 jinja

Jinja is Python based templating language modelled Django like templates, with modern look and designer friendly structure. Main specialties of Jinja is that, it is fast, widely used and secure with the optional sandboxed template execution environment. [92], [93]

Jinja has expessions similar to the Django framework but since it is based on Python, it provides python-like expressions while covering that the templates are evaluated in a sandbox. It is a text-based template language and thus can be used to generate any markup as well as source code. [93], [94]

Jinja is Flask's default template engine. This template engine also can allow refactoring of globals, filters, tests and tags. In addition to that, unlike Django template engine, it can enable templates to call functions for objects with proper arguments. [92]

Even though jinja is a hybrid language template based on Python and HTML with a pinch of JavaScipt, I decided to introduce jinja in frameworks sections since it aims to satisfy web based projects like BioNetVis.

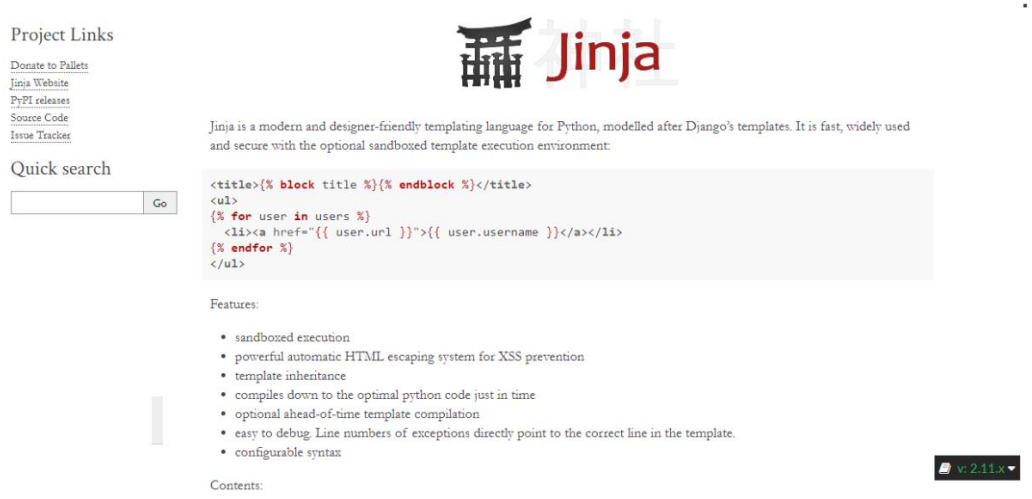

Figure 24: Jinja framework documentation and installation page [93]

3.2.3.3 Bootstrap

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. [95] It contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components. [96]

Main goal of Bootstrap is the unify design and give the consistent look of a web based project with freedom in respect of design and development aspects. Developers and designers can change layout, fonts, colors and size of each bootstrap elements to based their preferences. Once an element included to a project, Bootstrap gives the essentials interpretations for all HTML elements, hence the elements and page has a uniform look itself even there are many screen size for each device and browser rendering. On top of getting basic HTML, for styling aspects, developers can work with CSS classes which is predefined in Bootstrap as well.

Some of the JavaScript components comes in the form of jQuery elements. This ability gives developers extra tools to work on user interface. Every Bootstrap element consists basic HTML format, CSS expressions and occasionally JavaScript codes within. [96]

Most important part of the Bootstrap is that there are some essential elements in the framework, that affects entire web page without disrupting any other element in the source code. This ability is almost revolutionary for the front end development. One of the essential layout component is called ".container" and this element can save every other element wherever it is placed. Based on container type, it can adjust itself and the elements inside to the screen of the web-page showed on. Bootstrap has predefined class for heights and width for four types of screen sizes. There are follows as;

.xs : screen sizes less than 768px wide, mostly for phones

.sm : screens equal to or greater than 768px wide, for tables

.md : screens equal to or greater than 992px wide, for small laptops

.lg : screens equal to or greater than 1200px wide, for laptops and desktops [97]

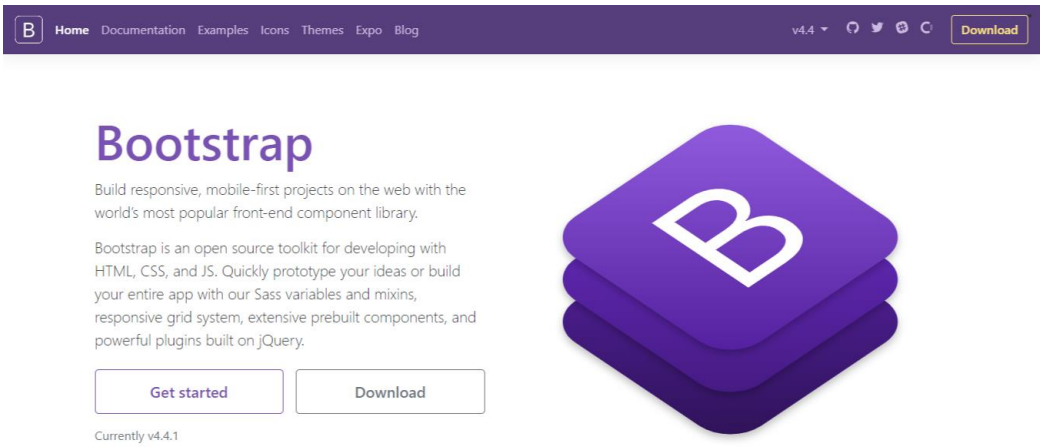

Figure 25: Bootstrap framework download and documentation page[95]

Among many features of Bootstrap, it's usability, responsive and uniform design, well documentation, ability to work with other frameworks and adversity of creating a web page with HTML, CSS and JavaScript from scratch, Bootstrap has been used while developing BioNetVis.

3.2.3.4 Sigma js

Sigma, referred as sigmajs, is a JavaScript library assigned to only graph drawing purposes. It makes easy to publish networks on Web pages, and allows developers to integrate network exploration in rich Web applications. [98]

Some of the main features of sigmajs is;

Custom rendering: Users can use customized rendering on top of other options such as Canvas or WebGL built-in renderers

Interactivity oriented: Events that will explained in the following chapters, hovering around the graphs, clicking and getting information from a node, zoom in, zoom out specialties.

Powerful graph model: Even though Sigma is just a rendering engine, events like visualization of customized graph algorithms is possible with customizations which in this work used.

Compatibility: Sigma can work on all of the modern browsers that can support Canvas, and WebGL, Additionally, it is faster on WebGL. [98], [99]

In visualization aspect, since user's need a tool that is interactive and informative while maintaining basic design principles, Sigmajs was the tool this project needed. Moreover, features like compatibility and ability to work with other frameworks made Sigma without a doubt, go to material of BioNetVis.

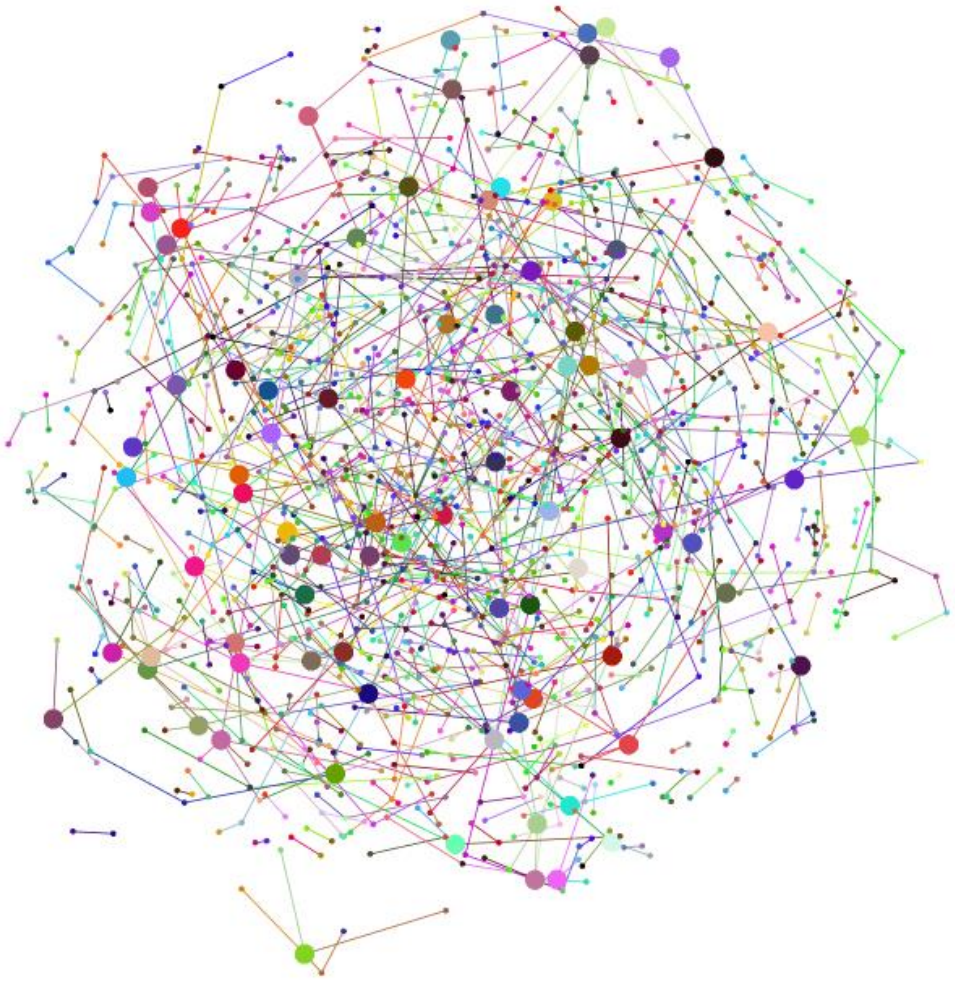

Figure 26: A Graph generated with sigma.js locally and 1000 nodes from the PPI Network data. Color and coordinates information generated randomly, size information is depended to connections. [98]

3.2.4 Libraries

In programming languages, in order to give meaningful commands to computer's processors, programmers use functions. Often a command can be used againg and again, recursively. To eliminate repatative and exhaustive coding, programmers can generalize these functions with methods. Like functions, methods also can be used repetitively. To deal with these problem programmers developed a solution called "programming libraries". These libraries can store many functions and method in order to work much efficiently and faster. Some of the libraries become essential even indispensable most of the cases. In this part we are going to present

libraries that have been used in order to develop BioNetVis. Later at section 3.3 we are take a look to the methods that has been used to generalize commands with their explanations as well.

Pandas: is an open-source and easy to use data analysis and manipulation tool and library that is integrated with python programming language. [100] With upda tes and recent developments especially in the data science aspect of the programmi ng, pandas becomes one of the essential and powerful library that needs to be le arn and use all the time. In addition to that more than 3000 pages' documentation and active support community, problem solving matters is much easier. [101] S ome of the key features ar

- Quick and powerful DataFrame object for handling data with efficient indexing
- Ability to reading and writing data with several data structres and variable formats such as text files, CSV files, Microsoft Excel and SQL databases and so on…
- Dynamic data alignment, handling missing data and organization from messy databases
- Reshaping and converting datasets
- Label-based dividing, subsetting and fancy indexing large datasets.
- Compiling and reconstructing data with group by feature, which can add split-apply-combine operations on datasets
- Powerful performance on merging and joining of data sets;
- Time series-functionality: date range generation and frequency conversion, moving window statistics, date shifting and lagging. Even create domain-specific time offsets and join time series without losing data;
- Optimized performance
- Python with pandas is in use in a wide variety of academic and commercial domains, including Finance, Neuroscience, Economics, Statistics, Advertising, Web Analytics, and more. [10

NumPy: is one of the essential library package in python for scientific computing and related works to that. [102] Among many other features, NumPy can be used as a reliable, multi-dimensional data container. Different data types can be defined on NumPy as well. This specialty is important since this thesis work based on organizing and visualization of the data itself. One of the main features are;

- Ability to contain N-dimensional array object
- It has connected and complex functions
- To be able to use linear algebra, fourier transform and many mathematical functions
- Tools for integrating C/C++ and Fortran code

Json – JSON encoder and decoder: JavaScript Object Notation (JSON) is a lightweight data interchange format inspired by JavaScript object literal syntax. [103] Basic usage on python is specified below.

```
json.dump(obj, fp, *, skipkeys=False, ensure_ascii=True, c
heck_circular=True, allow_nan=True, cls=None, indent=None,
 separators=None, default=None, sort_keys=False, **kw)
```

In addition to libraries, while developing BioNetVis, I created some objects to integrate with codes. These objects also will be explained in the programmed methods section 3.3 as well. Reader can also find the codes at the supplementary files at the end of the thesis work.

3.2.5 Network Data

Protein – Protein Interaction data have been taken from the supplementary files in the study "The Human Disease Network". [104] Goh et al. 's study investigates gene relations between know diseases thoroughly and gives results of the high quality laboratory experiments.

In order to create a network, we need some sort of information between elements. Goh et. al.'s dataset is basically a .sif file that contains Protein-Protein Interactions

between two proteins in each line end to be precise there are 62263 lines in the data file we have feed the implemented tool. Figure 25 shows the first and last ten of total twenty elements in the data set. For delimiter to separate elements from one another “pp” have been used. While creating a dataset without repetitive elements, this needed to be considered since some of the elements in the network has multiple neighbors. This logic explained the methods section.

	A	B		
1	TRIP6 pp PTK2		61254	ERCC2 pp CTDP1
2	MCM5 pp UBA52		61255	PSMA1 pp PSMD8
3	TRAF6 pp NUDT5		61256	CCBP2 pp CCL3
4	PAK2 pp MYH10		61257	TXNL4A pp PRPF4
5	TGFB1 pp TGFB1		61258	PIGS pp PIGT
6	TRAF2 pp FXR2		61259	EEF1A1 pp RPS6
7	AR pp UBE3A		61260	HTATIP pp HDAC1
8	RPL17 pp EIF2S1		61261	ROPN1 pp AKAP3
9	C20ORF172 pp CBX5		61262	RPS2 pp EIF3S6
10	HSPG2 pp NID1		61263	TRAF3 pp TNFRSF4

Figure 27: First and last ten elements of the data, with a delimiter as "pp"

3.3 Methods

In this section we are going to explain the approach to development of the project itself as well as the explanation of the written code with specified programming languages, frameworks and libraries mentioned in the previous chapters.

To achieve maximum efficiency in this thesis work and development of BioNetVis, waterfall development process used as cited in previously. With that being said, coding work divided into two chapters. These are; back-end approach, front-end approach. While we are going to explain reasoning and purpose of each line of code in these section, methods of this study also will revealed to the readers. We will conclude the chapter with a summary and gave use cases for the

project with finalized version of the BioNetVis. A discussion chapter will follow the conclude thesis work later with a reference page for sharing source codes of the project.

3.3.1 Back-End Approach

Back-end section is the backbone of the project itself. Without flexible and powerful back-end part the code of the part will be limited because of unchangeable mechanics. To prevent this limitations, I developed generalized methods which is compactable with any other datasets. In a result of that coding back-end section took much more time and energy but on the other hand the BioNetVis is become much more powerful visualization tool. In the below codes and purpose of each method or function is specified.

ReadCsv method is basic and self-explanatory method. This method allows program to read .csv files. Method requires a filename. A header is not necessary while executing this method.

```
def ReadCsv (filename, header=False):

    if not header:

        df=pd.read_csv(filename, header=None)

    else:

        df = pd.read_csv(filename)

    return df
```

ReadNetworkcsv method is reads network files whether these networks are small or large. With only changing delimeter section, any network file can uploaded to program to visualize. This generalization is the purpose of the back-end work. Not only for visualizing PPI Network, but also with any other network files as well. In this case “pp” is the delimiter that our network file uses. Like previous method, this one also requires a filename but not a header.

```
def readNetworkCsv(filename, header=False):
    if not header:
        df=pd.read_csv(filename, header=None, delimiter="pp")
```

```
    else:
        df = pd.read_csv(filename, delimiter="pp")
    return df
```

getUniqueColumns method is specified for the getting unique elements, in this case protein files. Otherwise repetitive data can distort the network. For this method to work, there has to be dataframe which is can be found in the codes.

```
def getUniqueColumns(dataframe):
    uniqArr=[]
    for col in dataframe.columns:
        localunique=dataframe[col].unique()
        # if len(uniqArr)==0:
        #     uniqArr = localunique[:]
        #     continue
        for localcol in localunique:
            if localcol not in uniqArr:
                uniqArr.append(localcol)
    return uniqArr
```

createNetworkDF is one of the important methods for this thesis work. To create a network and its' connection these methods has been created.

```
def createNetworkDF(dataframe):
    uniqueValues= getUniqueColumns(dataframe)
    print(len(uniqueValues))
    # dummyRow= [0 for i in range (0,len(uniqueValues))]
    df = pd.DataFrame()
    for value in uniqueValues:
        data={col:0 for col in uniqueValues}
        df= df.append(data,ignore_index=True)
    df.index = uniqueValues[:]
    return df
```

addNetworkConnection method adds elements to connection network. In order to this method to work, it requires a dataframe, a source and destination. Source and destination are the elements of the network that connected to each other. Notice, .format is python's relatively new feature that eliminates confusing code parts as well. This shows author's/developer's attention to lastest releases and dedicated effort to thesis work as well.

```
def addNetworkConnection(dataframe, src, dst):
    src_dst = dataframe.at[src, dst]
    dst_src = dataframe.at[dst, src]

    if src_dst == 0:
        dataframe.at[src, dst] = 1
    else:
        print ("Attempting to re-add {}-{} connection".format(src,
```

```
dst))

    if dst_src == 0:
        dataframe.at[dst, src] = 1
    else:
        print ("Attempting to re-add {}-{} connection".format(dst,
src))

    return dataframe
```

addNetworkCons is the method for adding connections to network. It is connected with .addnetworkConnection method. This method requires a dataframe and connections in order to work. There are definitions of source and destination as well dataframe. It reads connection values which is either 1 or 0 between elements, and adds to the network with previous method.

```
def addNetworkCons(dataframe, connections):
    for source, destination in connections.values:
        dataframe = addNetworkConnection(dataframe, src=source,
dst=destination)

    return dataframe
```

addProteinDataFrame This method automatically adds proteins to the dataframe and prevents proteins to be repetitive. If program reads an element, in this case protein, for the first time; program give the protein occurrence value 1 and adds it to the dataframe and updates the index which we will explain reasoning later.

```
def addProteinDataFrame(dataframe, protein, fromFile=None):
    data={"lines":protein, "occurences":1}

    if not fromFile == None:
        data['cluster'] = [fromFile]

    new_df = pd.DataFrame(data, index=[len(dataframe)])
    # print (new_df)
    dataframe =dataframe.append(new_df)
    dataframe = dataframe.reset_index()
    dataframe = dataframe.drop(columns=['index'])
    return dataframe
```

getProteinIndex is a self explanatory method as well. Program get's the protein index which is also protein name. This method is useful while finding connections between proteins.

```
def getProteinIndex(dataframe, protein):
    values = dataframe.index[dataframe["lines"]==protein]
    if len(values) == 0:
        return None
```

```
    else:
        return  values[0]
```

addOccurence method as name suggests adds occurrences to data frames. This method is used with front-end section, while user adds multiple proteins to the different clusters, if same element is in the different cluster this method allows to specifying which clusters has the element.

```
def addOccurence(dataframe, index, fromFile=None):
    # print (dataframe.loc[index]["lines"])
    # dataframe.loc[index] = {"lines": dataframe.loc[index]["lines"],
"occurences":dataframe.loc[index]["occurences"]+1}
    dataframe.at[index, "occurences"] += 1

    if not fromFile == None:
        clusterList = dataframe.at[index, 'cluster']
        if isinstance(clusterList, str):
            clusterList = [clusterList]

        if fromFile not in clusterList:
            clusterList.append(fromFile)
            dataframe.at[index, "cluster"] = clusterList
    # print (dataframe)
    return dataframe
```

iterateAndAppend is a generalized method for getting elements from .csv file to iterate. Elements that has not been exist in the dataframe will add, else if the element is already in the dataframe, this method calls .addOccurence method for adding occurrence to specific element, in this case proteins.

```
def iterateAndAppend(dataframe, csvDF, fromFile=None):
    for protein in csvDF.values:
        protein=protein[0]
        # print(dataframe.head())
        # print(protein)
        index = getProteinIndex(dataframe, protein)
        if index == None:
            dataframe = addProteinDataFrame(dataframe, protein,
fromFile)
        else:
            dataframe = addOccurence(dataframe, index, fromFile)
    return dataframe
```

searchExist is basic search method in the main dataframe. It is specifically written for indexing. While building connection network for PPI, it searches if and index is exists or not. This method gives the knowledge of a protein exists in the dataframe or not.

```
def searchExists(mainDataFrame, protein):
    searchIndex = mainDataFrame.index[mainDataFrame["lines"] ==
protein].tolist()
    if len(searchIndex) == 0:
        return None
    else:
        return searchIndex[0]
```

getAllConnections method as name suggest collects the information in the coonection network. getAllConnections enables the neighborhood properties. If there is a link between two proteins it is a valid connection and the respective adjacent matrix value between these proteins has value of 1. Otherwise it is not a valid connection and it appends the value of 0 to the matrix.

```
def getAllConnections(connectionDataframe, protein):
    columns = connectionDataframe.columns

    connections = []
    for column in columns:
        if connectionDataframe[column][protein] == 1.0:
            isValidConnection = True
        else:
            isValidConnection = False

        if isValidConnection:
            connections.append(column)
    return connections
```

listProteinsWithOccurenceX is basic filtering method to see proteins occurred with X times where X is an arbitrary number for user's input.

```
def listProteinsWithOccurencesX(DataFrame, X):
    proteins = []
    for index, row in DataFrame.iterrows():
        occurence = DataFrame.at[index, "occurences"]
        if occurence == X:
            proteins.append(DataFrame.at[index,"lines"])
    return proteins
```

For exemplary intentions and input code for .csv files and main data frame will be shown below as well. Full codes can be found on the supplementary file or source page of the thesis work.

```
Df1=ReadCsv(filename="path/to/your/file/lung?cancer_proteins.csv")
fileName="lung"
mainDataFrame=iterateAndAppend(dataframe=mainDataFrame, csvDF=df2,
fromFile=fileName)
print(mainDataFrame)
```

Node Object

In order to visualize proteins, we need to convert them as objects in order to work with the codes itself. This process requires creating a generalized object that will work with the respective framework, in this case, sigma.js.

For this purposes a node object is created while adding required information as well. Note that coloring part is leaved to be handled in the frond-end section since user will select the color of the added nodes to the cluster. This decision allows us to save from performance as well even though it seems unnecessary, note that Protein Protein Interaction Network parses 16480 protein to interact with each other meaning size of [16480x16480] matrix, 256000000 unit of element is created. Memory size and capacity of this network is simply extraordinary. To prevent memory loss and crashes sparse matrix is used.

```
class Edge:
    def __init__(self, source, target):
        self.id     = "e_{}_{}".format(source, target)
        self.source = source
        self.target = target
    def edgeToDict(self):
        data = {
            'id': self.id,
            'source': self.source,
            'target': self.target
        }
        return data
class Node:
    def __init__(self, protein, size=1):
        self.id     = protein
        self.label  = protein
        self.size   = size

    def setSize(self, size):
        self.size(size)

    def setSizeByMultiplier(self, occurence, size=None,
multiplier=None):
        if size == None:
            size = self.size
        if multiplier == None:
            multiplier = 1
        self.size = occurence * size * multiplier
    def nodeToDict(self):
        data = {
            'id': self.id,
```

```
            'label': self.label,
            'size': self.size
        }
        return data
```

Creating Json File

After creating an object, in order to create a json file that will interact with front-end part of the code, there has to be some work needs to be done. Code below gets the created nodes from the network and with the help of object, it adds node id, label, size with the respective edge information from the connection network.

```
def getEdges(dfnet,protein):
    # for column in dfnet.columns:
    indexes = list(dfnet.loc[dfnet[protein] == 1].index)

    return indexes
print (dfnet.columns)
for col in dfnet.columns:
    print ("\nprotein: {}\nedges: {}".format(col, getEdges(dfnet,
col)))

nodes = []
edges = []

for protein in proteinID:
    node = Node(protein)
    nodes.append(node)

    edgesList = getEdges(dfnet, node.label)
    for edgeValue in edgesList:
        edge = Edge(node.label, edgeValue)
        edges.append(edge)

jsonDict = {
    'nodes': [node.nodeToDict() for node in nodes],
    'edges': [edge.edgeToDict() for edge in edges]
}

jsonObject = json.dumps(jsonDict)

with open('data.json', 'w') as f:
    json.dump(jsonDict, f)
```

3.3.2 Front-End Approach

One of the keys for a successful story is the how to present it. Hence, the face of the BioNetVis has importance. While starting front-end approach this key was

important. Initially, work needed to be look much more professional than other web-based available tools for visualization yet easy to use. To achieve this task, other projects and ideas has been thoroughly inspected and decided to present BioNetVis as a single web page with additional information pages such as about and contact pages.

Boosting cognitive abilities of the user with single page and giving all the necessary information is intricate even though it seems easy. Interaction between user interface and user has to be understandable, changes should be visible quickly and every button's purpose on the page should be clear and has to work without a problem.

Reasons mentioned above, BioNetVis interface is reduced to be simple as much as possible while maintaining the high performance visualization tool behind the hood. Hood refers to back end and front-end section at the same time. In this subsection I will try to explain the coding parts of the front end section with the materials mentioned earlier. In a result of that, HTML, CSS, JavaScript sections will be explained with respective events, scripts and functions.

3.3.3.2.1 HTML Section

Home of the webpages are the faces presented information and because of that it is important as much as the codes behind the project. This section I will explain major coding parts of the front-end development.

This code block indicates the document we are starting is should be read as html file. In html code block there are some elements we need to indicate. These are head and body. In head section developers define the character set that will be used while writing code. Additionally, name of the page, reference to css code with stylesheet is also given. HTML codes has a feature to indicate starting and end points of the code blocks with "smaller than" and "greater than" signs. As you can see below these signs "< startofcode >" let compiler know this the starting point of that specific code block. To end that code block next to the smaller than sign, there is also a "slash" sign as well. "</ endofcode >"

We start codeblock with getting bootstrap framework inside of our project with taking stylesheet as well.

```
<!DOCTYPE html>
<html lang="en" >
<head>
  <meta charset="UTF-8">
  <title>BioNetVis 2.1</title>
  <link rel='stylesheet'
href='https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/4.4.1/css/bootstrap.min.css'><link rel="stylesheet"
href="./style.css">
```

In this code block we are starting to pull sigma.js framework and related plug-ins developed for sigma.

```
<body>
<head>
  <meta charset="UTF-8">
  <title>Test Page</title>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/sigma.js/1.2.0/sigma.min.j
s"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/sigma.js/1.2.0/plugins/sig
ma.parsers.json.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/sigma.js/1.2.0/plugins/sig
ma.renderers.edgeLabels.min.js"></script>
```

Bootstrap framework already has functions, methods we can use instead of creating by ourselves. This also eliminates time consuming part of the coding process. The codeblock below is for creating a navigation bar for the tool. This “navbar” contains other two pages for giving additional information about the thesis work.

```
<nav class="navbar navbar-expand navbar-light bg-light">
  <a class="navbar-brand" href="#">VizTool</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup"
aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
    <div class="navbar-nav">
      <a class="nav-item nav-link active" href="#">Home <span
class="sr-only">(current)</span></a>
      <a class="nav-item nav-link" href="#">About</a>
      <a class="nav-item nav-link disabled" href="#">Contact</a>
    </div>
```

```
  </div>
</nav>
```

Since the project is single page, there has to be some information needs to give quickly to server as a user manuel. Bootstrap has an element called “jumbotron” which gives highlighted information. Developers can add anything inside to a jumbotron and it can work smoothly. Additional grid system of bootstrap makes all of the elements responsive even though working with more than 16 thousand nodes on the tablets or phone is not recommended.

Code block below creates a jumbotron and we add a container to store information in it. In there title of the project and instructions are given with a “card” element as well. Instructions are deleted purposely for saving from the space on this page.

```
<div class="jumbotron jumbotron-fluid">
  <div class="container">
    <h1 class="display-4">VizTool</h1>
    <p class="lead">Biological network visualization based on protein-
protein interactions.</p>
    <div class="card">
      <div class="card-body">
        ...User Instructions...
      </div>
    </div>
  </div>
</div>
```

To use filtering elements on the project, radio buttons are used. To keep all of the filtering elements, respective section are combined in a column inside of a row and then a container as well. Similarly above, to save place ending section of the classes are not shown.

```
<div class="jumbotron jumbotron-fluid">
    <div class="container">
      <div class="row">
        <div class="col-3 my-col"> Filtering Elements
          <!--         Row 1 COL 1 FILTERING       -->
          <div class="row">
            Display HUMAN PPI Network:
            <div class="btn-group" data-toggle="buttons-radio">
              <button id="disppi_on" type="button" class="btn
active">On</button>
              <button id="disppi_off" type="button"
class="btn">Off</button>
            </div><br/> Show Common Proteins in the Clusters:

            (Occurences more than 2)
```

```
            <div class="btn-group" data-toggle="buttons-radio">
              <button id="disppi_on" type="button" class="btn
active">On</button>
              <button id="disppi_off" type="button"
class="btn">Off</button>
            </div><br/>
```

To post user's input into to BioNetVis, we need a input group. Input groups and folders are specifically designed for that. According to the user's input, PPI Network updates itself and changing respective colors. There are six input groups, text boxes there.

```
</div>
        <div class="col-9 my-col">Text Boxes
          <div class="row">
            <!--           TB1 -->
            <div class="col">
              <div class="input-group">
                <div class="input-group-prepend">
                  <input id="cluster_form" type="text" style="font-
weight: bold;" value="Cluster 1"/>
                </div><textarea resize: none class="form-control
textarea" aria-label="With textarea"></textarea></div>
            </div>
            <!-- TB2-->
            <div class="col">
              <div class="input-group">
                <div class="input-group-prepend">
                  <input id="cluster_form" type="text" style="font-
weight: bold;" value="Cluster 2"/>
                </div><textarea class="form-control textarea" aria-
label="With textarea"></textarea></div>
            </div>
```

Since PPI Network has no coordinates assigned for the proteins, all of the nodes are given random coordinates. And this system makes hard to understand the data and the visualization. To overcome this problem we come up with a strong solution. Creating an function and event trigger to function that will pull all of the nodes that are neighbor to each other and based on the density on the canvas, move nodes to much empty spaces so that there won't be any tangled network. . ForceAtlas allows is created for this. Since network is refreshing with all the new coordinates, ForceAtlas needs to be turned off manually, otherwise it is constantly works. To trigger ForceAtlas, a toggle button is created and added on top of the canvas for sigma and visualization.

```
<div class="custom-control custom-switch">
  <input type="checkbox" class="custom-control-input"
```

```
id="customSwitches" onchange="isToggle()">
  <label class="custom-control-label" for="customSwitches">Toggle
ForceAtlas</label>
</div>
```

An export button for network built by BioNetVis, as .svg format is also created. Snap() event is called from JavaScript codes, which will be shown after HTML Section.

```
<li onclick="snap()" class="list-group-item"><button  type="button"
class="btn btn-dark"  >export</button></li>
```

Sigma framework is based on JavaScript language, yet to show it we need to get help from HTML. Sigma is called in the codes below in a container. That container created a workspace, called canvas, to visualize network on the canvas like paintings. To give information about nodes and clicked events, Information panel added also below the canvas.

```
<!-- sigmaJS visualization -->
<div class="container container-fluid"><div id=sigma-
container></div></div>
<!-- NODE INFO PANEL <-->
<div class="container">
  <div class="jumbotron jumbotron-fluid">NODE INFO PANEL
    </li>
  <li class="list-group-item" >Names:
    <p id="nodeName"><p> </li>
</div>
</div>
```

Latter for giving citing information we conclude HTML part with a jumbotron and Bootstrap's finalizing codes which will be given in the thesis' github page.

3.3.3.2.2 JavaScript

JavaScript part of the front-end equally important while comparing html section due to main event, required information stored, processed and visualized in here. We start JS with defining the data. Note that this data created from back-end section of the project. There are more than 16000 nodes in the data and for each note there are id, label, edges, size, coordinates, color information stored in the data. For saving space purposes only dummy data showed below.

```
var data ={  "nodes": [    {
      "id": "n0",
      "label": "A node",
      "x": 0,
      "y": 0,
      "size": 3
    },
    {
      "id": "n1",
      "label": "Another node",
      "x": 3,
      "y": 1,
      "size": 2
    },
    {
      "id": "n2",
      "label": "And a last one",
      "x": 1,
      "y": 3,
      "size": 1
    }
  ],
  "edges": [
    {
      "id": "e0",
      "source": "n0",
      "target": "n1"
    },
    {
      "id": "e1",
      "source": "n1",
      "target": "n2"
    },
    {
      "id": "e2",
      "source": "n2",
      "target": "n0"
    }
  ]
}
```

PPI Data does not contain any coordinates information, as we mentioned above. For visualizing data and interact with the elements, proteins, this information needs to be created. Code block showed below is creates and assigns each node x and y coordinates on the canvas. In addition to that, instead of changing color of each node manually, automating this feature we added one line of code in this function as well.

```
// function to produce random xy position for each node
function xyCoordinate (data){
```

```
  let len = Object.keys(data.nodes).length
  let nodes = data.nodes
  for(let i=0;i<len;i++){
    nodes[i].x = Math.random();
    nodes[i].y = Math.random();
    nodes[i].size -= Math.random();
    nodes[i].color = '#'+(Math.random()*0xFFFFFF<<0).toString(16);
  }
}
xyCoordinate(data)
```

To initiate sigma with existing data, a container is created on the sigma framework. Note that this container is called on the html part.

```
// the graph function
s = new sigma({
  graph: data,
  container: 'sigma-container',
  settings: {
    scalingMode: 'outside',
},
});
```

Click and get neighbors method and function. Pretty self-explanatory as title indicates, this method returns an information each time when it is called. Later we use this information for highlighting the nodes event. Neighbors also showed it in the information panel which we call on the html part.

```
// function to get the neighbor nodes
sigma.classes.graph.addMethod('neighbors', function(nodeId) {
  var k,
      neighbors = {},
      addn = {}
      index = this.allNeighborsIndex[nodeId] || {};
  for (k in index)
    neighbors[k] = this.nodesIndex[k];
   // for (k in neighbors)
  //   addneighbors[k] = this.allNeighborsIndex
  // neighbors[k] = neighbors.concat(addneighbors[k]);
  return neighbors;
});
```

This built-in function in the sigma.js highlights the color of clicked node and its neighbors. While this event happens, rest of the unselected nodes are passive and becoming invisible to user.

```
// function to color only the clicked node and its neighbours
function picker(s) {

  s.graph.nodes().forEach(function(n) {
    n.originalColor = n.color;
 //         if the node has nieghbours give it a bigger size
        if (Object.keys(s.graph.neighbors(n.id)).length>2){
          n.size = 0.5
        }
        else{
          n.size = 0.1
        }
      });
      s.graph.edges().forEach(function(e) {
        e.originalColor = e.color;
      });


  s.bind('clickNode', function(e) {
    var nodeId = e.data.node.id,
        toKeep = s.graph.neighbors(nodeId);
    toKeep[nodeId] = e.data.node;
    document.getElementById("nodeName").innerHTML =
Object.keys(toKeep);

    s.graph.nodes().forEach(function(n) {
      if (toKeep[n.id])
        n.color = n.originalColor;
      else
        n.color = '#eee';
    });

    s.graph.edges().forEach(function(e) {
      if (toKeep[e.source] && toKeep[e.target])
        e.color = e.originalColor;
      else
        e.color = '#eee';
    });
    s.refresh();
  });


  s.bind('clickStage', function(e) {
    s.graph.nodes().forEach(function(n) {
      n.color = n.originalColor;
    });

    s.graph.edges().forEach(function(e) {
      e.color = e.originalColor;
    });

    // Same as in the previous event:
    s.refresh();
```

```
  });
}
picker(s)

To toggle ForceAtlas, code block below is used.

var t = true
function isToggle(){
  if (t){
    s.startForceAtlas2(
      {worker: true, barnesHutOptimize: false,
       scalingRatio:10,
       adjustSizes: true

      })
  }else{
    s.stopForceAtlas2()

  }
  t=!t

}
```

To export created visualization a renderer added from sigma's plugins and later with snap() function it is called and downloaded to user's device as .svg format.

```
//export button script
// Adding a canvas renderer
s.addRenderer({
  container: 'graph-container',
  type: 'canvas'
});
function snap() {
  console.log('exporting...');
  var output = s.toSVG({download: true, filename: 'mygraph.svg', size:
1000});
  // console.log(output);
};
```

Color picker function is required in order to get user's desired color selection for the specific cluster and visualize on the network. For that a modern bootstrap color picker have been used. [105]

```
// Colorpicker
(function ($) {
   $('#cp_1, #cp_2, #cp_3, #cp_4, #cp_5,
#cp_6').colorpicker().on('colorpickerChange colorpickerCreate',
function (e) {
      $(this).siblings().css('border-color', e.value);
      $(this).children('.input-group-text').css('border-color',
e.value);
```

```
    });
})(jQuery);
```

3.4 Proposed Method

In this part we will explain the proposed BioNetVis with its features and usability for the user. Like above, before each figure an explanation will be made with its reasoning.

Like every web-based tool, our purpose is give user a simple, understandable instructions with easy to use user interface. Figure 28 shows the header of the proposed work with several information. Elements in the footer are; a header to give basic explanation of the BioNetVis, a hero unit which highlights the information above it and simple instructions for researchers to use tool.

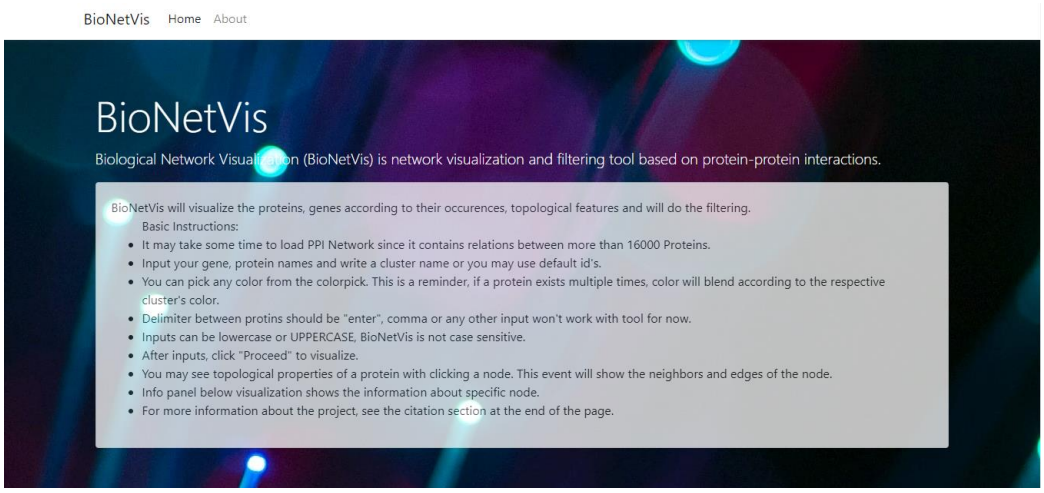

Figure 28: BioNetVis' Landing Page

Figure 29 shows the input area of the proposed tool for user the log their genes or proteins based on their intentions to use the BioNetVis. Cluster names and the respective colors next to input areas are predefined and can be changed with clicking on them. There for the visualizing input nodes user might select the color, it’s density and also, for future references rgb values are shown as well.

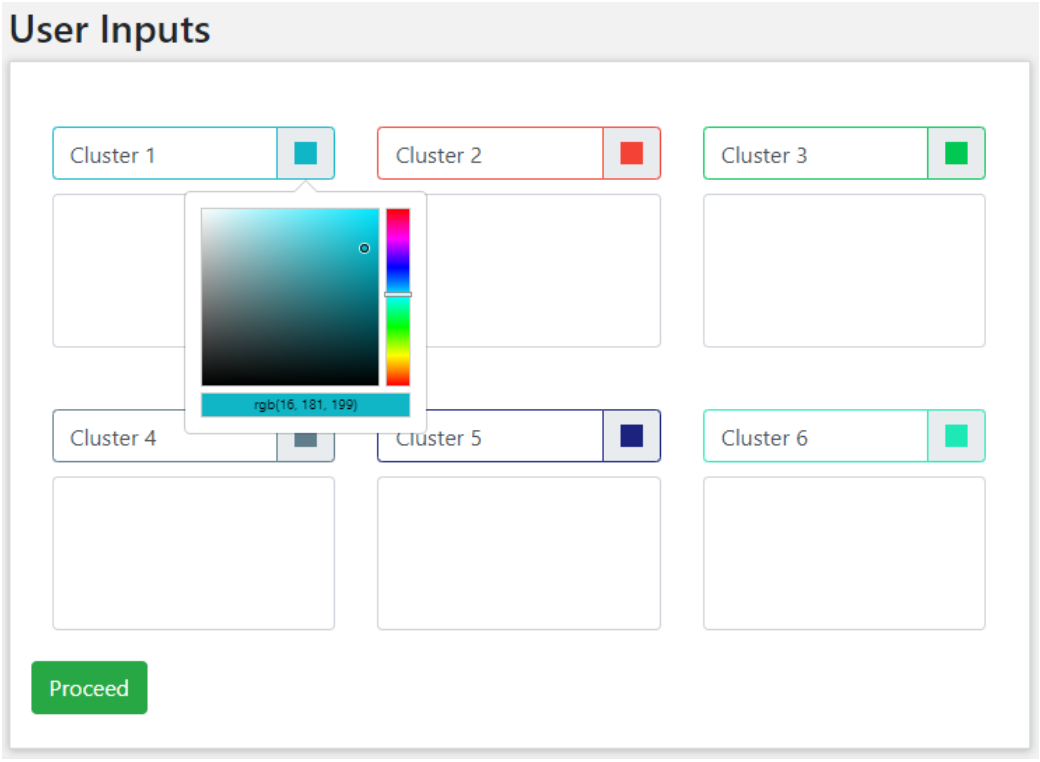

Figure 29: Input areas and color selection event of BioNetVis

Visualizing cannot be complete without filtering based on the topological properties of the nodes. These properties have been used while visualizing based on proteins interactions with neighbor numbers and applied to the node's size. Moreover, some of the filtering elements are depended on the user's input. Hence simple explanations have been made next to them. There user can find filtering options for displaying on or off Human PPI network. Since user input does not contain any relation about connections within, BioNetVis cannot visualize randomly solo nodes, as a result of these option will not work if user did not add any elements and can be found in the network. Occurrences, showing common proteins and searching features are simple and self-explanatory filtering options. Figure 30 shows the filtering elements of the work.

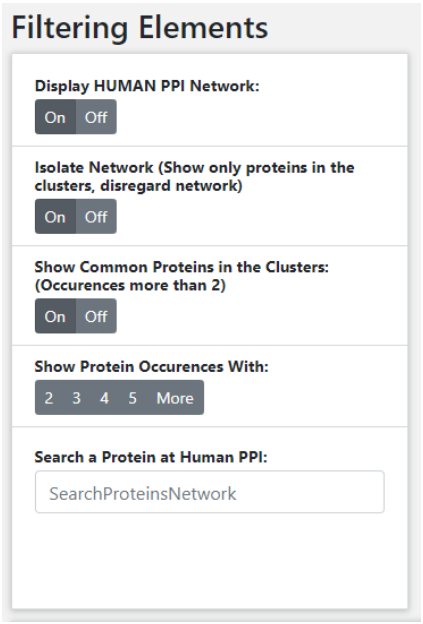

Figure 30: Filtering elements of the BioNetVis

BioNetVis visualize PPI Network nodes randomly and for that reason visualizing more than 16 thousand nodes is visually tiring and requires a lot of computational power. To explain simplistically and showing the work that BioNetVis can handle visualization have been made with 1300 nodes for this proposed tool section. Figure 31 shows a zoomed out image of visualizing. All of the nodes' coordinates are randomly created and assigned. Based on nodes topological properties size of the nodes changes. Node labels cannot be seen without hovering on them or zooming in. More visualization will be shown.

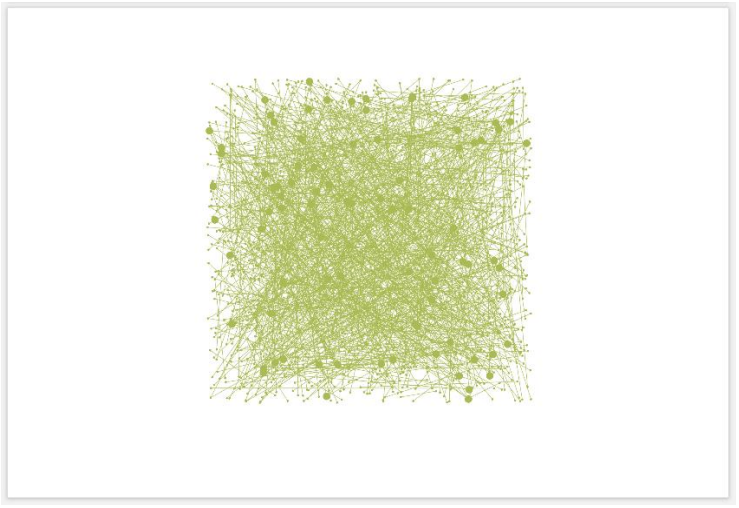

Figure 31: A simple visualization example with BioNetVis

Figure 32 shows the snap event while clicking a node, the clicked node and its neighbor nodes are highlighted and keep their original color. Every other node in the visualization is changing color and remains passive until clicking an empty coordinate on the canvas.

Figure 32: Highlighting a node and neighbors event

Randomly visualized nodes are not the suitable option for visualizing and understanding the data. To overcome this problem a built-in plugin, ForceAtlas have been used with the sigmajs. ForceAtlas has ability to change coordinates and based on their relations. Since it is an intuitive problem, ForceAtlas updates the coordinates constantly. That means if user does not end the work of ForceAtlas, it will be work continuously. Hence to on and off ForceAtlas a toggle button has been placed on top of visualization panel, that triggers ForceAtlas. Figure 33 shows ForceAtlas toggle button and export button that will downloads visualization in .svg format.

Figure 33: ForceAtlas toggle button and export button

Figure 34 and 35 respectively shows the output results while toggling ForceAtlas and inputting some proteins and selecting a color.

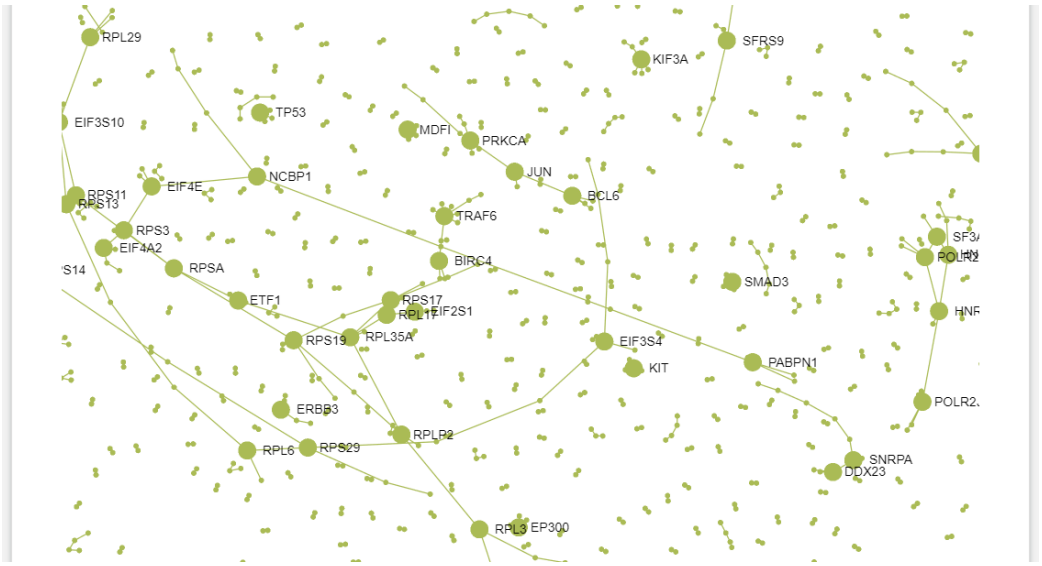

Figure 34: Toggling ForceAtlas for 3 seconds. It continuously updated the coordinates and pulls the nodes that interacts with each other.

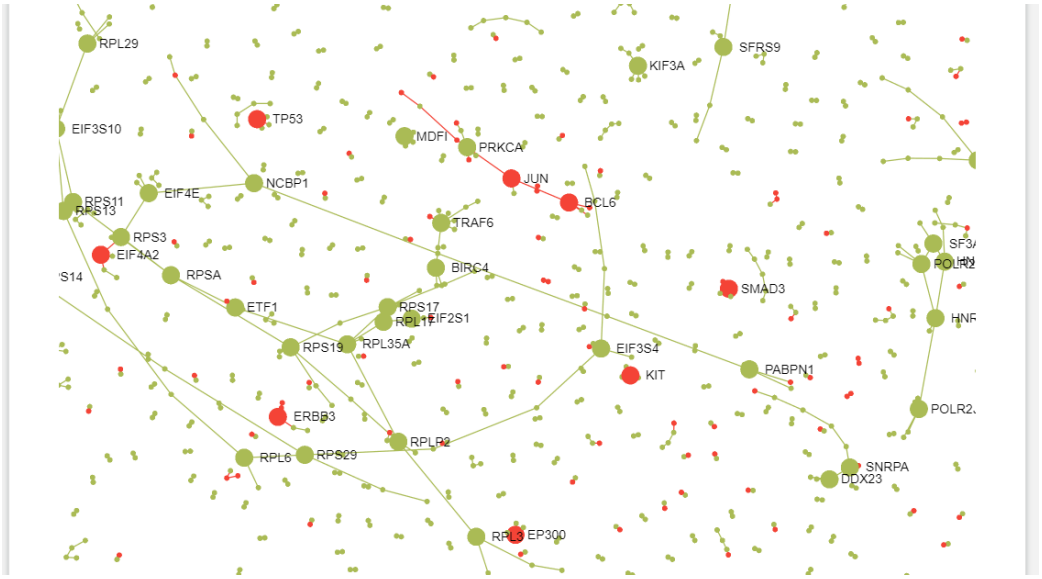

Figure 35: After adding arbitrary protein names to cluster 2 and visualizing nodes.

To highlight common proteins in the clusters, color blending technique have been used. This technique is mostly have been used for two colors in the programming. Luckily in this project we improve this technique for more than two colors. BioNetVis can mix more than two colors. Figure 36 shows color of a randomly selected node “kit” added three different clusters and each time color has been

changed. This feature is one of the important specialization for computational drug repurposing and highlighting important nodes, genes.

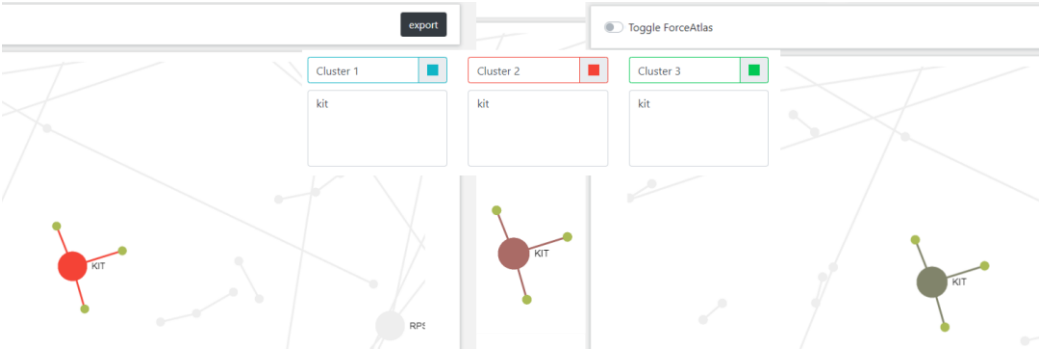

Figure 36: Multiple color mixing feature. (4 layer of image is combined to save up space) From left to right. A randomly selected protein "kit" has been added Cluster 2 and visualized. This processed followed after "kit" added and visualized with cluster 1 & 2. Lastly, it visualized with adding protein to cluster 1, 2 & 3.

Figure 37 shows node information panel, that after clicking a node, neighbor nodes of that specific node and cluster that has been inputted can be seen.

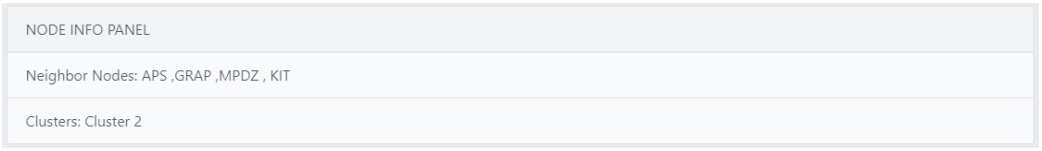

Figure 37: Node information panel of BioNetVis

Last element of proposed work is citation panel for giving proper citation information to the user. Even though currently this thesis work is written, BioNetVis has great potential and will be updated regularly. For raising awareness to the tool, a paper is also in the works and will be placed here after publication.

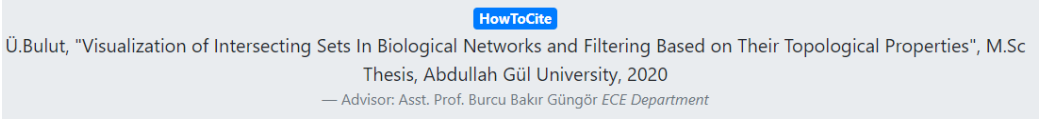

Figure 38: Citation panel for the BioNetVis

Chapter 4

Use Case

In this chapter we are going to present a use case for the developed thesis. Measuring success of a tool is subjective and relatively distant from the project itself. Most of the implementation thesis, projects, papers do not contain a results section. Instead many of them relates their work with “usability” of the implemented project. Hence, in this chapter to relate with results we are going to present a use case for BioNetVis.

Use cases are the milestones for the implementation projects. For our project it is clear to say, a working tool for aimed tasks can be considering as success. After developing the BioNetVis with several aims that mentioned in the introduction, such as drug repurposing, personal drug research, rare diseases treatment and investigation, tool’s testing, scaling for online environment and optimizing the written code itself is also challenging. Tackling this tasks and visualization with specific purpose, let’s say a drug repurposing research would come out via BioNetVis, thesis work achieved what it aims and thus makes it a successful.

Before showing use case, to show capacity of the project, there are couple points should be mentioned. In terms of scalability, visualizing a PPI Network online with more than 16000 nodes is incredible achievement if we consider the information this network stores. With adding user’s input data to visualizing, it needs to be a strong tool in the background and BioNetVis meets this needs. In addition to that, elegant and simple user interface with flexible features like compatibly to the many devices such as phones, tablets and desktops, makes the BioNetVis much more compelling when we consider it’s non-competitive similar tools.

Lastly, another reason for to say this thesis is succeeded is that besides a working tool itself, the person who tried to develop this project has many endless nights and tried to learn couple of programming languages with some other bioinformatics background as well. I hope this is enough.

And this use case result will show us that developing a web based tool like BioNetVis for bioinformatics eliminates some of the downsides of package based programs and give additional benefits. It has power of complex package programs in a web interface with simple design. Proposed tool is a perfect candidate to take places of simple visualization tools. This use case is merely a surface of the program that we developed and researchers and academics can be used.

4.1 Drug Repurposing for Alzheimer's and Cerebral Palsy

In this use case we are going to implement a drug repurposing research into the BioNetVis. The main focus on the drug repurposing is studying two diseases commonly occurred protein and found target genes. Afterwards this targeted genes are searched in the drug's target genes datasets. Moreover, if targeted genes are using in the another disease, that is the starting point of the computational drug repurposing. Hence we need two disease information, mainly name of related genes and targeted drug. BioNetVis will come to the stage here. Instead of focusing tedious researching, data discovery and cleaning, finding occurrences between these two disease becomes too much effort for this simple task, all user have to do is simply adding gene values to the text fields in the BioNetVis. The tool itself automatically updates the visualization and show occurrences clearly in the network. Later we focus on the targeted genes, drugs and repurposing of this specific drug.

For this specific use case obviously the data needed. Currently, there are some projects for storing bioinformatics data and publishing online. For convenience, as data source we are going to use DisGeNET.

DisGeNET is a research site which contains one of the largest databases for genes and variants related to the human diseases open to the public. [106]–[109]. DisGeNET includes new data with an expert curation repository, Genome-Wide Association Studies (GWAS) catalogues, animal models and scientific publications. Collected data is homogeneously annotated and controlled vocabulary used for general community. [109]

Last version of DisGeNET is currently v6.0 contains 628,685 gene-disease associations (GDAs), between 17,549 genes and 24,166 diseases, disorders, traits, and clinical or abnormal human phenotypes, and 210,498 variant-disease associations (VDAs), between 117,337 variants and 10,358 diseases, traits, and phenotypes. [108], [109] For more data collection process you can refer cited articles and web pages. [106], [107], [110]–[112]

For use case collected genes are for Alzheimer's diseases and Cerebral Palsy diseases taken from DisGeNET. In addition to that since BioNetVis is need only gene names collected data cleaned from unnecessary information.

BioNetVis can visualize more than 16000 Nodes without giving errors. Unfortunately, due to insufficient graphics cards unit on many computers, computation takes time and often it canvas element of visualization crashes while trying to compiling data. As you can see in the figure 39, our graphics cards unit cannot handle this amount of data. We need to remind that with each time an event triggered such as zooming in and out, clicking a node to highlight neighbors and edges, all of nodes edges and nodes themselves are compiling again. In result of that we couldn't show the commonly affected proteins and find targeted drug. Additionally, we can say BioNetVis is capable of visualizing larger amount of nodes and networks. But to work with this amount of information requires a lot of computational power. With higher performance hardware units this issue can be resolved.

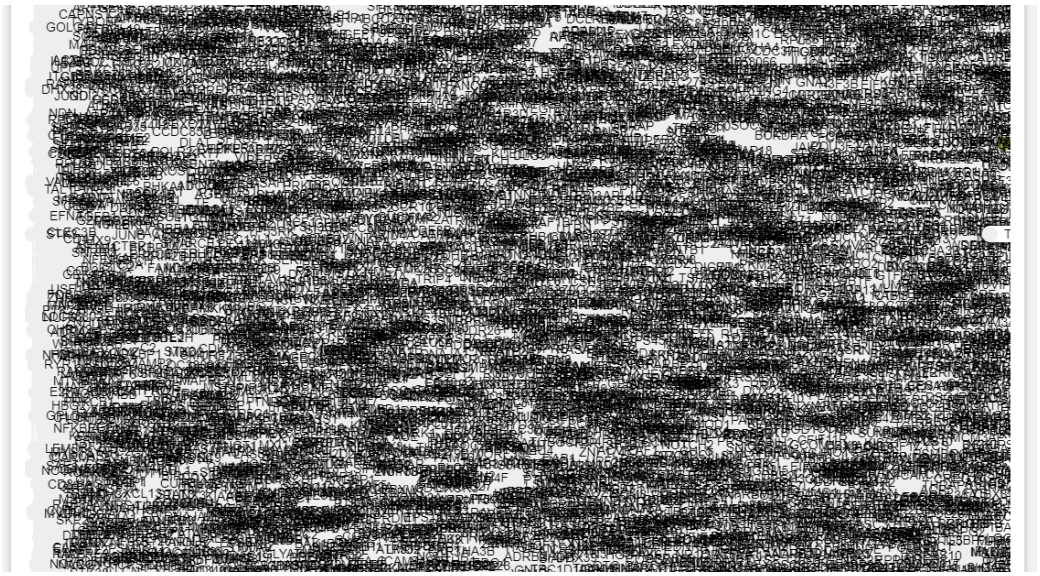

Figure 39: After visualizing for a split second due to insufficient graphics power unit canvas cannot properly visualize and only visualizing element crashes due to Sigma Framework.

Chapter 5

Conclusions and Future Prospects

5.1 Conclusions

Scientific researchers aim to improve human life and knowledge with available tools and widen it as much as possible trying new perspectives. This is true for any academician, or a researcher that they motive their study while they start to a new scientific paper, project, thesis work or a similar study. Likewise, when I start this study, my aim was not to change the world with a brand new invention or tool, but just to improve it as much as I can. BioNetVis, which is the proposed tool in this thesis is the "current" end result for now.

There are several tools that await their turn to be used and they are disregarded for unknown reasons. Most of the times, publicity or even design can effect the tool's success. The main advantage of BioNetVis is simply its usability among geneticists, bioinformaticians, scientific researchers and developers. It is placed in between complex package-based tools and simple, elementary grade visualization tools.

Can it be improved further? Of course like every web-based project, BioNetVis is also designed and developed in this era's necessities and requirements. In future, it could be outdated in terms of its design and capabilities. There is still some room for the improvement for BioNetVis.

This thesis work has its own limitations in terms of time and work-force. Since it is developed with one people under Thesis advisor's supervision, there can be that needs to be improved later on. In the following section, we will indicate potential reference points and possible improvements as our future work. These future

prospects can make the tool more suitable to a much broader audience and more powerful.

5.2 Future Prospects

In this section, we are going to address future prospects and plans to improve the improvement aspects of BioNetVis. Before listing the future plans for BioNetVis, there are a couple of questions that needs to be answered. These are related with other approaches for this specific project and if there are, the restrictions needed to be lifted.

There are other approaches that can be followed while building a tool like BioNetVis. Because of the current available tools, these approaches can be seen as unnecessary for several reasons. For example, adding complex features for other bioinformatics tasks will require prior knowledge and memory. Since it's an online tool migrating from online to creating a package based tool would come as a weak option because of its competitor's strong motives, industry level work force and behavior. On the other hand, BioNetVis define itself as easy-to-use interface with strong background coding execution process. Making it much more simplistic is basically a step backwards.

In terms of restrictions, BioNetVis has some limitations. It has several framework dependencies for visualization and front-end sections. Creating libraries from scratch, and changing visualization framework to a new tool becomes time-consuming and similar event to try to invent wheel again.

In despite of that some changes can be acceptable and actually move further to BioNetVis. While we add these suggestions there will be two main focus areas; design wise and coding wise.

Design is abstract and related to human's taste, hence it constantly moves. Yet we can always add little features to increase "likeability" of the tool and engage with users. In this work since I was learned JavaScript later, interaction between sigma canvas can be enhanced in couple ways. One adding bigger canvas would be

helpful. Secondly information panel for the visualization is currently very simple. Adding more events and animations can help understanding the visualization. In addition to that, DOM Operations can be added to make interface much smoother. Additionally, currently once user adds information, if nodes exists only color and density of the node changes. Instead of color change, changing the node's structure to a pie chart would be much more efficient to see for a specific node in which cluster it belongs to.

In terms of coding wise automation of the BioNetVis definitely can be improved. Currently the tool uses one main network which is PPI network. At the back-end codes, there is already a method created for changing visualized bigger network. If user want to upload it's own network that is possible with back-end codes. However, since codes already has some computational challenges this feature is disabled and will be activated for future releases of BioNetVis. Additionally, changing node's structure also requires writing code process as well.

To conclude, BioNetVis is already a powerful and flexible tool for visualization, and biological data analysis, that could be potentially used in drug repurposing and rare disease treatment studies. Yet, it can be and will be enhanced in the future as planned above. It is obvious that BioNetVis has its own limitations yet it is build for one specific purpose. The tool does not claim that it will change the world, it is simply a representation of a humble idea that each individual can contribute to improve our environment.

BIBLIOGRAPHY

[1] F. Ruskey and M. Weston, “A Survey of Venn Diagrams,” *THE ELECTRONIC JOURNAL OF COMBINATORICS*, 1997. [Online]. Available: https://www.combinatorics.org/files/Surveys/ds5/ds5v3-2005/VennEJC.html. [Accessed: 17-Mar-2020].

[2] V. K. Pounraja, G. Jayakar, M. Jensen, N. Kelkar, and S. Girirajan, “A machine-learning approach for accurate detection of copy number variants from exome sequencing,” *Genome Res.*, vol. 29, no. 7, pp. 1134–1143, 2019.

[3] J. Thomas and K. Cook, *Illuminating the Path: Research and Development Agenda for Visual Analytics*. IEEE-Press, 2005.

[4] S. O. Sümer, “CHISIO WEB : A WEB-BASED FRAMEWORK FOR CUSTOMIZABLE VISUALIZATION OF RELATIONAL INFORMATION,” 2012.

[5] D. W. Mount, “Bioinformatics : Sequence and Genome Analysis . Second Edition. By David W Mount . Bioinformatics : Sequence and Genome Analysis . Second Edition . by David W Mount The Quarterly Review of Biology , Vol . 80 , No . 1 (March 2005), p . 109,” vol. 80, no. 1, pp. 51–52, 2014.

[6] “Bioinformatics - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Bioinformatics. [Accessed: 14-Mar-2020].

[7] I. Y. Abdurakhmonov, “Bioinformatics : Basics , Development , and Future,” pp. 3–28, 2016.

[8] A. M. Lesk, “Bioinformatics | science | Britannica,” *Encyclopædia Britannica, inc.* [Online]. Available: https://www.britannica.com/science/bioinformatics. [Accessed: 13-Mar-2020].

[9] “Omics - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Omics. [Accessed: 14-Mar-2020].

[10] “DNA - Wikipedia.” [Online]. Available:

https://en.wikipedia.org/wiki/DNA. [Accessed: 14-Mar-2020].

[11] A. Mashaghi and A. Katan, “A physicist’s view of DNA,” Nov. 2013.

[12] “WHO | WHO definitions of genetics and genomics.” [Online]. Available: https://www.who.int/genomics/geneticsVSgenomics/en/. [Accessed: 14-Mar-2020].

[13] “Genomics.” [Online]. Available: https://en.wikipedia.org/wiki/Genomics. [Accessed: 14-Mar-2020].

[14] “RNA - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/RNA. [Accessed: 14-Mar-2020].

[15] “RNA | Definition, Structure, Types, & Functions | Britannica.” [Online]. Available: https://www.britannica.com/science/RNA. [Accessed: 14-Mar-2020].

[16] “Transcriptome - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Transcriptome. [Accessed: 14-Mar-2020].

[17] W. P. Blackstock and M. P. Weir, “Proteomics: Quantitative and physical mapping of cellular proteins,” *Trends Biotechnol.*, vol. 17, no. 3, pp. 121–127, 1999.

[18] N. L. Anderson and N. G. Anderson, “Proteome and proteomics: New technologies, new concepts, and new words,” *Electrophoresis*, vol. 19, no. 11, pp. 1853–1861, 1998.

[19] R. L. Anderson, Johnathon D.; Johansson, Henrik J.; Graham, Calvin S.; Vesterlund, Mattias; Pham, Missy T.; Bramlett, Charles S.; Montgomery, Elizabeth N.; Mellema, Matt S.; Bardini, “Comprehensive Proteomic Analysis of Mesenchymal Stem Cell Exosomes Reveals Modulation of Angiogenesis via Nuclear Factor-KappaB Signaling,” pp. 601–613, 2016.

[20] “Human Genome Project .” [Online]. Available: https://en.wikipedia.org/wiki/Human_Genome_Project. [Accessed: 11-May-2020].

[21] S. (Battelle M. I. Tripp and M. (Battelle M. I. Grueber, “Economic Impact of the Human Genome Project,” 2011.

[22] G. Vaidyanathan, “Redefining clinical trials: The age of personalized

medicine," *Cell*, vol. 148, no. 6, pp. 1079–1080, 2012.

[23] "Metabolite - Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Metabolite. [Accessed: 14-Mar-2020].

[24] "Metabolomics - Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Metabolomics. [Accessed: 14-Mar-2020].

[25] B. Daviss, "Growing Pains for Metabolomics," *The Scientist*, 2005. [Online]. Available: https://www.the-scientist.com/technology/growing-pains-for-metabolomics-48835. [Accessed: 14-Mar-2020].

[26] T. T. Ashburn and K. B. Thor, "Drug repositioning: Identifying and developing new uses for existing drugs," *Nat. Rev. Drug Discov.*, vol. 3, no. 8, pp. 673–683, 2004.

[27] A. Sertkaya, Aylin; Birkenbach, "Examination of clinical trial costs and barriers for drug development," *Erg*, pp. 679–715, 2011.

[28] Y. Yeu, Y. Yoon, and S. Park, "Protein localization vector propagation: a method for improving the accuracy of drug repositioning," *Mol. Biosyst.*, vol. 11, no. 7, pp. 2096–2102, 2015.

[29] H. Xue, J. Li, H. Xie, and Y. Wang, "Review of drug repositioning approaches and resources," *Int. J. Biol. Sci.*, vol. 14, no. 10, pp. 1232–1244, 2018.

[30] S. Pushpakom *et al.*, "Drug repurposing: Progress, challenges and recommendations," *Nat. Rev. Drug Discov.*, vol. 18, no. 1, pp. 41–58, 2018.

[31] A. Breckenridge and R. Jacob, "Overcoming the legal and regulatory barriers to drug repurposing," *Nat. Rev. Drug Discov.*, vol. 18, no. 1, pp. 1–2, 2018.

[32] M. Zitnik, F. Nguyen, B. Wang, J. Leskovec, A. Goldenberg, and M. M. Hoffman, "Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities," *Inf. Fusion*, vol. 50, no. June 2018, pp. 71–91, 2019.

[33] D. M. Gysi *et al.*, "Network Medicine Framework for Identifying Drug Repurposing Opportunities for COVID-19," 2020.

[34] R. Kerber, “Old drugs, new life / Boston Globe,” *Boston Globe Newspaper*, 2003. [Online]. Available: http://archive.boston.com/business/globe/articles/2003/12/31/old_drugs_new_life/. [Accessed: 05-May-2020].

[35] O. Osakwe and S. A. A. Rizvi, *Social Aspects of Drug Discovery, Development and Commercialization*. Academic Press, 2016.

[36] A. P. Kumar, S. Lukman, and M. N. Nguyen, *Drug repurposing and multi-target therapies*, vol. 1–3. Elsevier Ltd., 2018.

[37] J. L. Medina-Franco, J. Yoo, and A. Dueñas-González, “DNA Methyltransferase Inhibitors for Cancer Therapy,” *Epigenetic Technol. Appl.*, pp. 265–290, 2015.

[38] J. Jesús Naveja, A. Dueñas-González, and J. L. Medina-Franco, *Drug Repurposing for Epigenetic Targets Guided by Computational Methods*. Elsevier Inc., 2016.

[39] S. H. Sleigh and C. L. Barton, “Repurposing strategies for therapeutics,” *Pharmaceut. Med.*, vol. 24, no. 3, pp. 151–159, 2010.

[40] J. W. Astin *et al.*, “Innate immune cells and bacterial infection in zebrafish,” *Methods Cell Biol.*, vol. 138, pp. 31–60, 2017.

[41] B. A. Kidd and J. T. Dudley, *Systems Immunology*. Elsevier Inc., 2016.

[42] H. Heberle, V. G. Meirelles, F. R. da Silva, G. P. Telles, and R. Minghim, “InteractiVenn: A web-based tool for the analysis of sets through Venn diagrams,” *BMC Bioinformatics*, vol. 16, no. 1, pp. 1–7, 2015.

[43] J. C. (Centro N. de B. (CNB-C. Oliveros, “Venny 2.1.0.” [Online]. Available: https://bioinfogp.cnb.csic.es/tools/venny/. [Accessed: 17-Mar-2020].

[44] H. Chen and P. C. Boutros, “VennDiagram: a package for the generation of highly-customizable Venn and Euler diagrams in R,” *BMC Bioinformatics*, 2011.

[45] V. Nagarajan and M. Pirooznia, “GeneVenn - a web application for comparing gene lists using Venn diagrams. .” [Online]. Available: http://genevenn.sourceforge.net/. [Accessed: 17-Mar-2020].

[46] T. Hulsen, J. de Vlieg, and W. Alkema, “BioVenn - A web application for the comparison and visualization of biological lists using area-proportional Venn diagrams,” *BMC Genomics*, vol. 9, pp. 1–6, 2008.

[47] T. Hulsen, J. de Vlieg, and W. Alkema, “BioVenn - a web application for the comparison and visualization of biological lists using area-proportional Venn diagrams,” *2008*. [Online]. Available: http://www.biovenn.nl/. [Accessed: 18-Mar-2020].

[48] “Medical Systems Biology - VennMaster.” [Online]. Available: https://sysbio.uni-ulm.de/?Software:VennMaster. [Accessed: 18-Mar-2020].

[49] H. A. Kestler *et al.*, “VennMaster: area-proportional Euler diagrams for functional GO analysis of microarrays.,” *BMC Bioinformatics*, vol. 9, p. 67, 2008.

[50] H. A. Kestler, A. Müller, T. M. Gress, and M. Buchholz, “Generalized Venn diagrams: A new method of visualizing complex genetic set relations,” *Bioinformatics*, vol. 21, no. 8, pp. 1592–1595, 2005.

[51] P. Bardou, J. Mariette, F. Escudié, C. Djemiel, and C. Klopp, “venn: an interactive Venn diagram viewer,” *BMC Bioinformatics*, vol. 15, no. 293, pp. 1–7, 2014.

[52] “jQuery.” [Online]. Available: https://jquery.com/. [Accessed: 18-Mar-2020].

[53] “Graph and Data Visualization | Tom Sawyer Software.” [Online]. Available: https://www.tomsawyer.com/graph-and-data-visualization/. [Accessed: 03-May-2020].

[54] H. Cai *et al.*, “VennPlex-A Novel Venn Diagram Program for Comparing and Visualizing Datasets with Differentially Regulated Datapoints,” *PLoS One*, vol. 8, no. 1, 2013.

[55] B. Martin *et al.*, “VENNTURE-A novel Venn diagram investigational tool for multiple pharmacological dataset analysis,” *PLoS One*, vol. 7, no. 5, 2012.

[56] M. E. Smoot, K. Ono, J. Ruscheinski, P. L. Wang, and T. Ideker,

"Cytoscape 2.8: New features for data integration and network visualization," *Bioinformatics*, vol. 27, no. 3, pp. 431–432, 2011.

[57] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks. BT - International AAAI Conference on Weblogs and Social," *Int. AAAI Conf. Weblogs Soc. Media*, pp. 361–362, 2009.

[58] "NetworkX — NetworkX documentation." [Online]. Available: https://networkx.github.io/. [Accessed: 06-May-2020].

[59] "NodeXL: network analysis & insights as easy as pie charts." [Online]. Available: https://nodexl.com/. [Accessed: 06-May-2020].

[60] "Graphviz - Graph Visualization Software." [Online]. Available: https://www.graphviz.org/. [Accessed: 06-May-2020].

[61] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers, "The State-of-the-Art of Set Visualization," *Comput. Graph. Forum*, vol. 35, no. 1, pp. 234–260, 2016.

[62] J. R. Conway, A. Lex, and N. Gehlenborg, "UpSetR: An R package for the visualization of intersecting sets and their properties," *Bioinformatics*, vol. 33, no. 18, pp. 2938–2940, 2017.

[63] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot, H. Pfister, and A. Manuscript, "UpSet: Visualization of Intersecting Sets Europe PMC Funders Group," *IEEE Trans Vis Comput Graph*, vol. 20, no. 12, pp. 1983–1992, 2014.

[64] A. Lex and N. Gehlenborg, "Points of View (37): Sets and intersections," *Nat. Methods*, vol. 11, no. 8, p. 779, 2014.

[65] L. Wilkinson, "Exact and approximate area-proportional circular venn and euler diagrams," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 2, pp. 321–331, 2012.

[66] "hms-dbmi/UpSetR: An R implementation of the UpSet set visualization technique published by Lex, Gehlenborg, et al.." [Online]. Available: https://github.com/hms-dbmi/UpSetR. [Accessed: 06-May-2020].

[67] Ü. Bulut, "Biological Network Visualization (BioNetVis),

umitbulut/BioNetVis.” [Online]. Available: https://github.com/umitbulut/BioNetVis. [Accessed: 20-Jun-2020].

[68] Python Software Foundation, “Python.org.” [Online]. Available: https://www.python.org/. [Accessed: 02-Apr-2020].

[69] T. E. Oliphant, “Python for Scientific Computing Python Overview,” *Comput. Sci. Eng.*, pp. 10–20, 2007.

[70] D. Kuhlman, “A Python Book,” *A Python B.*, pp. 1–227, 2013.

[71] Python Software Foundation, “PyPI · The Python Package Index.” [Online]. Available: https://pypi.org/. [Accessed: 04-Apr-2020].

[72] “JavaScript - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/JavaScript#cite_note-tc39-7. [Accessed: 04-Apr-2020].

[73] “ECMAScript® 2021 Language Specification.” [Online]. Available: https://tc39.es/ecma262/#sec-overview. [Accessed: 04-Apr-2020].

[74] D. Flanagan, *JavaScript: The Definitive Guide, Sixth Edition*. O’Reilly Media, 2011.

[75] D. R. Brooks, *Programming in HTML and PHP*. 2017.

[76] “HTML - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/HTML. [Accessed: 04-Apr-2020].

[77] W3C, “‘What is CSS?’ World Wide Web Consortium.” [Online]. Available: https://www.w3.org/standards/webdesign/htmlcss#whatcss. [Accessed: 04-Apr-2020].

[78] “Cascading Style Sheets - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Cascading_Style_Sheets. [Accessed: 04-Apr-2020].

[79] “Source-code editor - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Source-code_editor. [Accessed: 05-Apr-2020].

[80] “PyCharm: the Python IDE for Professional Developers by JetBrains.” [Online]. Available: https://www.jetbrains.com/pycharm/. [Accessed: 05-Apr-2020].

[81] “Collaboration with Anaconda, Inc. | PyCharm Blog.” [Online]. Available: https://blog.jetbrains.com/pycharm/2019/04/collaboration-with-anaconda-inc/. [Accessed: 05-Apr-2020].

[82] “Apache License, Version 2.0.” [Online]. Available: https://www.apache.org/licenses/LICENSE-2.0. [Accessed: 05-Apr-2020].

[83] “PyCharm Community Edition and Professional Edition Explained: Licenses and More | PyCharm Blog.” [Online]. Available: https://blog.jetbrains.com/pycharm/2017/09/pycharm-community-edition-and-professional-edition-explained-licenses-and-more/. [Accessed: 05-Apr-2020].

[84] “PyCharm - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/PyCharm. [Accessed: 05-Apr-2020].

[85] “Full-stack Web Development - Features | PyCharm.” [Online]. Available: https://www.jetbrains.com/pycharm/features/web_development.html. [Accessed: 05-Apr-2020].

[86] “CodePen: Build, Test, and Discover Front-end Code.” [Online]. Available: https://codepen.io/. [Accessed: 05-Apr-2020].

[87] “About CodePen.” [Online]. Available: https://codepen.io/about/. [Accessed: 05-Apr-2020].

[88] “Welcome to Flask — Flask Documentation (1.1.x).” [Online]. Available: https://flask.palletsprojects.com/en/1.1.x/. [Accessed: 06-Apr-2020].

[89] “Flask | The Pallets Projects.” [Online]. Available: https://palletsprojects.com/p/flask/. [Accessed: 06-Apr-2020].

[90] “Flask (web framework) - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Flask_(web_framework). [Accessed: 06-Apr-2020].

[91] “Extensions — Flask Documentation (1.1.x).” [Online]. Available: https://flask.palletsprojects.com/en/1.1.x/extensions/. [Accessed: 06-Apr-2020].

[92] “Jinja (template engine) - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Jinja_(template_engine). [Accessed: 06-Apr-2020].

[93] “Jinja — Jinja Documentation (2.11.x).” [Online]. Available: https://jinja.palletsprojects.com/en/2.11.x/. [Accessed: 06-Apr-2020].

[94] “Jinja | The Pallets Projects.” [Online]. Available: https://palletsprojects.com/p/jinja/. [Accessed: 06-Apr-2020].

[95] “Bootstrap · The most popular HTML, CSS, and JS library in the world.” [Online]. Available: https://getbootstrap.com/. [Accessed: 06-Apr-2020].

[96] “Bootstrap (front-end framework) - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework). [Accessed: 06-Apr-2020].

[97] “Grid System · Bootstrap · Migrating to v4.” [Online]. Available: https://getbootstrap.com/docs/4.4/migration/#grid-system. [Accessed: 06-Apr-2020].

[98] “Sigma js.” [Online]. Available: http://sigmajs.org/. [Accessed: 06-Apr-2020].

[99] “Home · jacomyal/sigma.js Wiki.” [Online]. Available: https://github.com/jacomyal/sigma.js/wiki. [Accessed: 06-Apr-2020].

[100] “pandas - Python Data Analysis Library.” [Online]. Available: https://pandas.pydata.org/. [Accessed: 20-Apr-2020].

[101] W. P. D. T. McKinney, “pandas: powerful Python data analysis toolkit Release 1.0.3,” *Pydata Organization*. [Online]. Available: https://pandas.pydata.org/docs/pandas.pdf. [Accessed: 20-Apr-2020].

[102] “NumPy — NumPy.” [Online]. Available: https://numpy.org/. [Accessed: 20-Apr-2020].

[103] “json — JSON encoder and decoder — Python 3.8.2 documentation.” [Online]. Available: https://docs.python.org/3/library/json.html#rfc-errata. [Accessed: 20-Apr-2020].

[104] K. Il Goh, M. E. Cusick, D. Valle, B. Childs, M. Vidal, and A. L. Barabási, “The human disease network,” *Proc. Natl. Acad. Sci. U. S. A.*,

vol. 104, no. 21, pp. 8685–8690, 2007.

[105] “Bootstrap Colorpicker, a color picker component for jQuery, compatible with Twitter Bootstrap.” [Online]. Available: https://itsjavi.com/bootstrap-colorpicker/. [Accessed: 09-May-2020].

[106] J. Piñero *et al.*, “DisGeNET: A discovery platform for the dynamical exploration of human diseases and their genes,” *Database*, vol. 2015, pp. 1–17, 2015.

[107] J. Piñero *et al.*, “DisGeNET: A comprehensive platform integrating information on human disease-associated genes and variants,” *Nucleic Acids Res.*, vol. 45, no. D1, pp. D833–D839, 2017.

[108] J. Piñero *et al.*, “The DisGeNET knowledge platform for disease genomics: 2019 update,” *Nucleic Acids Res.*, vol. 48, no. D1, pp. D845–D855, 2020.

[109] “DisGeNET - a database of gene-disease associations.” [Online]. Available: https://www.disgenet.org/home. [Accessed: 28-Apr-2020].

[110] A. Bauer-Mehren, M. Rautschka, F. Sanz, and L. I. Furlong, “DisGeNET: A Cytoscape plugin to visualize, integrate, search and analyze gene-disease networks,” *Bioinformatics*, vol. 26, no. 22, pp. 2924–2926, 2010.

[111] A. Bauer-Mehren, M. Bundschus, M. Rautschka, M. A. Mayer, F. Sanz, and L. I. Furlong, “Gene-disease network analysis reveals functional modules in mendelian, complex and environmental diseases,” *PLoS One*, vol. 6, no. 6, 2011.

[112] “DisGeNET - a database of gene-disease associations.” [Online]. Available: https://www.disgenet.org/dbinfo. [Accessed: 28-Apr-2020].