Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Prediction of Phenotype from Structural Genomic Variations

Bachelor Thesis

2020
Kristína Balažovičová

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Prediction of Phenotype from Structural Genomic Variations

Bachelor Thesis

Study Programme:	Bioinformatics
Field of Study:	Computer Science and Biology
Department:	Department of Computer Science
Supervisor:	Mgr. Werner Krampl

Bratislava, 2020
Kristína Balažovičová

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Kristína Balažovičová
Študijný program: bioinformatika (Medziodborové štúdium, bakalársky I. st., denná forma)
Študijné odbory: informatika
biológia
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Prediction of Phenotype from Structural Genomic Variations
Predikcia fenotypu zo štrukturálnych genomických variantov

Anotácia: Moderné sekvenačné metódy umožnili štúdium variability ľudského genómu a jej dopadu na vonkajšie znaky jedinca (fenotyp). Jedným zo základných typov variabilít sú štrukturálne zmeny súvislých úsekov genómu (tzv. štrukturálna variabilita), ako je zduplikovanie alebo vynechanie časti genómu. Cieľom práce je preskúmať možnosť predikcie dopadu takýchto zmien genómu fenotyp, ako je zdravie a vzhľad jedinca. Práca predpokladá použitie verejne dostupných dát, ich špecifické spracovanie, návrh vhodného klasifikátora a vhodnú vizualizáciu pre praktické použitie v klinickej diagnostike.

Vedúci: Mgr. Werner Krampl
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: prof. RNDr. Martin Škoviera, PhD.

Dátum zadania: 29.10.2019

Dátum schválenia: 29.10.2019 doc. Mgr. Bronislava Brejová, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname:	Kristína Balažovičová
Study programme:	Bioinformatics (Joint degree study, bachelor I. deg., full time form)
Field of Study:	Computer Science, Informatics Biology
Type of Thesis:	Bachelor´s thesis
Language of Thesis:	English
Secondary language:	Slovak

Title: Prediction of Phenotype from Structural Genomic Variations

Annotation: Modern sequencing methods allowed research of human genome variability and its impact on external traits of human individual (so-called phenotype). One of basic types of variabilities are structural genomic changes (so-called structural variations) such as duplications or deletions of genomic segments. Aim of this thesis is to assess the impact of these genomics changes on the phenotype such as individual's health and appearance. Student will use publicly accessible data, apply domain specific processing, and design suitable classifier and visualization for practical use in clinical diagnostics.

Supervisor:	Mgr. Werner Krampl
Department:	FMFI.KI - Department of Computer Science
Head of department:	prof. RNDr. Martin Škoviera, PhD.
Assigned:	29.10.2019

Approved: 29.10.2019 doc. Mgr. Bronislava Brejová, PhD.
Guarantor of Study Programme

.....
Student

.....
Supervisor

Acknowledgments: I would like to thank my supervisor Mgr. Wernew Krampl for his help, willingness and patience. I am also grateful to my family for their support.

Abstrakt

Štruktúrne varianty sú zmeny v štruktúre DNA, ktoré môžu ovplyvňovať vzhľad a zdravie jedinca, súhrne označované ako fenotyp. Zaoberali sme sa možnosťami predikcie fenotypu ľudského plodu z dát získaných sekvenovaním jeho DNA. Vvytvorili tri predikčné modely, ktoré využívajú dáta o štruktúrnych variantoch z verejne dostupných zdrojov. Popísali sme spôsob úpravy týchto dát a taktiež ako proces vzniku modelov. Tiež sme porovnali a popísali výsledky modelov na našich dátach.

Kľúčové slová: fenotyp, predikcia, štruktúrne varianty.

Abstract

Structural variations are changes in structure of DNA and they can influence health and appearance of individual, which is referred to as phenotype. We were focused on possibility of prediction of phenotype of human fetus from the data obtained by DNA sequencing. We created three prediction models, which use data of structural variations from publicly available sources. We described the way this data were adjusted and how the models were created. We compared and described results of our models.

Keywords: phenotype, prediction, structural variations

Contents

List of Figures

List of Tables

Introduction

Nowadays, public interest in DNA sequencing rapidly increased. Next generation sequencing technologies has made possible to obtain large amount of genomic data for an affordable price. These data has wide utilization in many different scientific fields and in clinical practice. This technology enables extensive analyses of human genome and the variation in its structure as well.

Variation in structure of DNA can have significant impact on phenotype of individual. They can cause serious genetic diseases or symptoms, however, structural variations with no negative effect are known as well.

Throughout years the diseases and syndromes linked to the specific structural variations were assembled and stored in various publicly available databases.

In this thesis we attempted to use data from these databases to predict potential effect of variation, which occurred in DNA of an individual, on its phenotype. In present days, when the structural variation in the genome of a fetus is detected, it is difficult to predict whether it would have negative influence on its health and appearance or it would be harmless. The aim of this work is to create prediction models, which could predict possible impact of the structural variation on the phenotype of the fetus.

We created three model based on different approaches, which will use different sources of data. But all of the three model make their prediction based on the records of clinically observed relations between changes on human genome and the caused phenotypes.

In the first chapter we describe and explained the changes in DNA, mutations and structural variants in detail. We acquaint the reader with classes of mutations, their influence and importance and their origin. We mention the characteristics of individual classes of mutations as well. The first chapter contains a detailed description of CNV, due to this type of structural variations is the center of our interest.

The second chapter introduce data sources, which were used in this thesis, as well as the programming language and various packages designed for this language.

In the third chapter we describe the adjustment of obtained data and the process of creating our three models. In the end of this chapter we provide several statistic, which evaluate the results of our models.

Chapter 1

Genomic variations and phenotypes

Years back, the cost of DNA sequencing used to be significantly higher. However, with new technologies emerging, the price of DNA sequencing is more affordable and, naturally, it has found its usage in different medical fields. Besides other usages, it is also used for prediction of possible fetus's syndromes. By sequencing the DNA of fetus, several types of mutations might be detected with serious effects on the fetus's phenotype . Despite the fact that there are mutations that have no observable manifestation, many of known mutations have significant negative influence on the life of individual, even may be lethal. In this chapter we describe various types of changes in human genome, explain how do they arise in genetic information and illustrate what influence and importance they have.

1.1 Mutations

A mutation is broad term covering extensive quantity of modifications in section of DNA [7]. In general, these modifications of genome may be caused by exposure to radiation, UV light or exposure to ionizing radiation. However, they may be also caused by errors in processes of DNA replication, mitosis (cell division, which results in two identical cell with retained number of chromosomes) and meiosis (cell division process, which produces haploid gametes as sperm or egg cells), processes of DNA repair or by other types of DNA damage.

Since mutations may be lethally dangerous and there are several opportunities for them to arise, the cell has mechanisms to check and repair resulting damages in the DNA. This mechanisms are designed to search and correct errors in the DNA with the aim to prevent arising mutation and to return the damaged sequence to proper state. However, not all of emerged DNA errors are successfully repaired and thus some of them give rise to new mutations.

1.1.1 Impact of mutations

Mutations, which occur in genome of organism, may affect sections of various lengths. The number of influenced bases may vary from one extreme to another, as there are lot of known mutations changing the DNA in only single base (for example single-nucleotide polymorphism or one base deletion or insertion) as well as mutations with a size of a whole chromosome of organism (for example, aneuploidy). Naturally, there are mutations and changes in DNA of any size besides mentioned lengths and the length of mutation usually have certain influence on significance of mutation.

Additionally, impact and effect induced by mutation have strong connection with a part of genome where the mutation occurred. Phenotypes of individuals differ in dependence of which genes or regions of DNA are affected by the mutations. A good example are Mendelian disorders. Mendelian disorders are genetic disorders caused by mutation in a single gene. For illustration, we can take achondroplasia (one of syndromes is dwarfism) and oculocutaneous albinism (which is certain type of albinism). Both are diseases caused by mutation in only one gene, but phenotypes they lead to are significantly different. [11] [31]

1.1.2 Importance and hleritability of mutations

Besides congenital syndromes (such as Down syndrome) and genetic diseases which aggravate the life of an individual, mutations also significantly affect whole populations of organisms. Mutations are responsible for arising of new genes and together with genetic recombination gives rise to genetic variability and subsequently, new phenotypes (new colors of eyes, type of hair), which are essential for evolution. Although in most cases mutations cause negative changes, which decrease fitness of an individual, certain part of mutations do have neutral or even positive result on ability to survive and reproduce [34]. As a consequence of natural selection, it may happen that this mutations prevail in population and subsequently, accumulation of mutations can lead to a formation of new species. Thus mutations play very important role in evolution and also in diversity within the species.

But not all of gene mutation are transferred to next generations. Since mutations are classified by inheritance to hereditary (inheritable) and somatic (only in somatic cells) mutations, only the hereditary mutations can be under selection. Changes in DNA are caused by hereditary mutations, which occur in germ cells - sperms or eggs. After conception of organism, these mutations are present in all of its cells. On the other hand, somatic mutations are mutations that are induced by outer factors, as exposure to radiation or chemical substances, thus occurring during the lifetime of an individual. Naturally, they appear in only a portion of organism's cells and because of this, they cannot be transferred to organisms's offspring unless they are present in

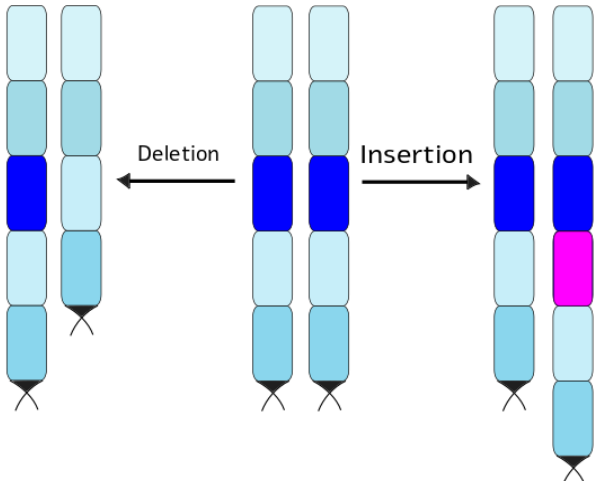

Figure 1.1: This figure displays both events - deletion on the left and insertion on the right side of the reference sequence in middle of the picture.

germ cells [33] [32].

Apart from classification by inheritance, we divide mutations by way which variation they cause. In general, genetic variations are divided into three categories, which are indels (insertion and deletions), single nucleotide variations and structural variations [8].

1.2 Indels

The term indel is an abbreviation for names of two genetic variations - insertions and deletions. Category of indels includes only small insertions and deletions with length varying in the range from two base pairs to a few hundreds [8]. Example of both of them is shown on Figure 1.1. The indel mutations are the second most copious type of variation in humans and the majority of them is caused by slippage of polymerase during DNA replication [27]. When indels occur in coding sequence of human genome, they are likely to cause frameshift mutation.

Frameshift mutation is a change in coding sequence of genome and is caused by adding or deleting bases in number which is not divisible by three. Coding sequences consist of triplets of nucleotides creating codons, corresponding to triplet genetic code. As soon as is this arrangement changed by indel indivisible by three, whole information is shifted and resulting protein coded by this new mutated code will possibly have very different qualities in comparison to original protein.

1.3 Single nucleotide variations

Single nucleotide variation (SNV) is mutation in a single nucleotide and represents substitution of one base on specific position in genome by another base. Figure 1.2 [5] illustrates single nucleotid variation between two strands of DNA. Single nucleotide variations are the most frequent type of DNA alternation in humans [4]. They are most commonly present in noncoding regions of DNA and most of these do not have any negative effect on phenotype of the individual. These occur in human genome normally and are part of variability within populations. Although there are many harmless single nucleotide variations, SNVs can also occur in coding regions of genome or in regulatory regions of genes. Their presence there may significantly affect the expression of the gene or the type of protein which is coded by this gene. However, due to impact of selection, which has often much stronger effect on SNVs in coding and regulatory regions than on other SNVs, there is observable disproportion between them in human genome [35].

As was mentioned above, single nucleotide variation is the most frequent genetic variation in humans. According to the reference genome, average healthy human has approximately from four millions to five millions genomic variations. More than 99% of them is represented by single nucleotide variantions and short indels, which means that in average, there is one SNV or indel in almost every kilobase of human genome, according to reference genome [4].

When specific base on specific position in genome differs from reference, but is present in more than 1% of the population, it is termed single nucleotide polymorphism. Many of SNPs are known and are associated with certain symptoms or reaction to drugs, although a lot of them do not cause disfunctions itself, but in a combination with other alternations of DNA.

1.4 Genomic structural variations

Structural variations represents modification in structure of a chromosomal DNA. According to general definition, structural variations are sections of chromosomal DNA with altered number of copies, orientation or location in chromosome. The alternations in number of copies cover insertions and deletions. Both of them fall under the term copy number variations. When a change in orientation occurs, we refer to it as inversion. The term translocation is used in relation to transitioned location. Part of structural variations, named balanced, do not induce any change in amount of genetic material. The opposite, unbalanced structural variations, are connected to gain or loss of parts of DNA.

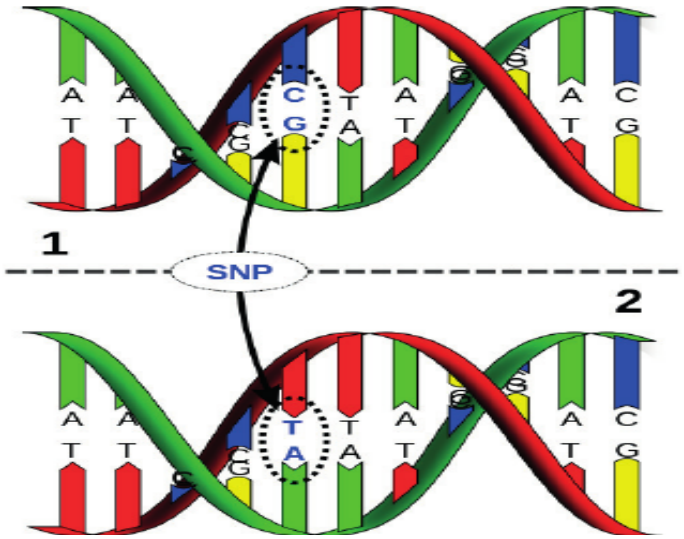

Figure 1.2: In this figure is displayed illustration of Single nucleotide polymorphism between two samples of DNA strand.

1.4.1 Inversions

Part of balanced structural variations are inversions. Inversion cause overturning of section of chromosomal DNA. After this modification, whole inverted section is placed on chromosome without added or missing bases or other genetic material, but in reversed order to the original sequence. Inversions are located on the same chromosome as was the section before overturning, they do not relocate any genetic material between more chromosomes.

Two types of inversions are distinguished - paracentric inversions and pericentric inversions. They differ in location within the chromosome, exactly in relation to centromere. The first type, paracentric inversions occur in regions which do not include the centromere of chromosome. On the other hand, the second type, pericentric inversion cover sections of chromosome containing the centromere. Figure 1.3 shows simple example of both the types.

Inversion usually do not cause any pathological phenotype, unless the break in chromosome is placed directly in a gene splitting it into two parts. After overturning this section, the gene stays broken and completely non-functional. If the gene with some vital function is affected by this inversion, this mutations is lethal for the individual [14].

1.4.2 Translocations

In the term translocations are included balanced and unbalanced translocations that belong to balanced and unbalanced structural variations respectively. Both types of

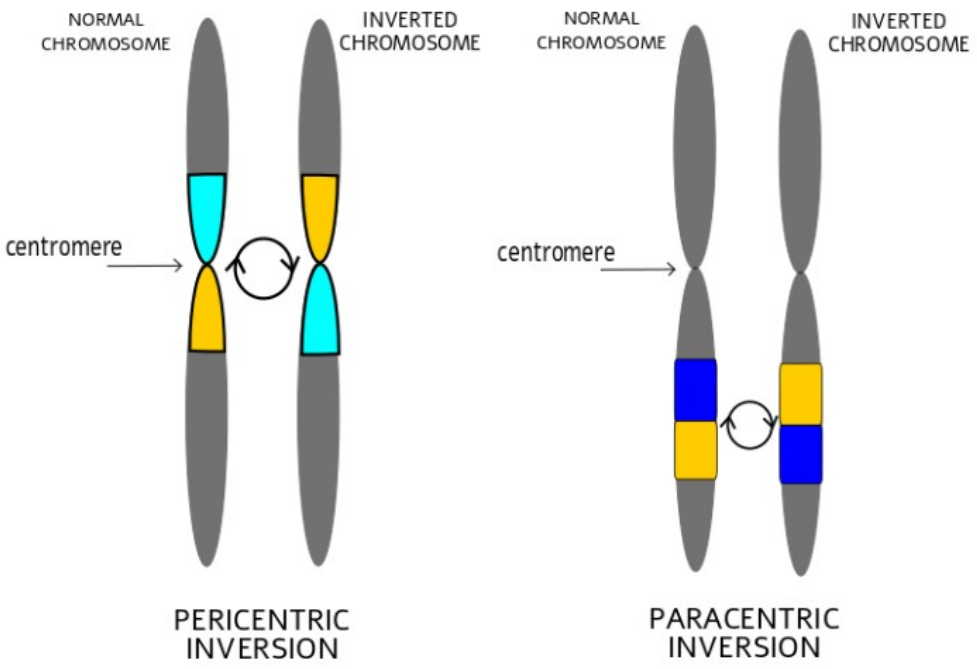

Figure 1.3: This figure shows example of both types of inversion. The main difference between paracentric and pericentric inversion is very obvious here.

translocations cause exchange of sections of two chromosomes. We provide illustrative example in the Figure [20]. This can generate two derived chromosomes, where, in the case of balanced translocations, each of them contains complete and undamaged section of the other one. The different case, unbalanced translocations, also lead to exchange of sections between two chromosomes, but these section do not have to be found on the other chromosome entire and intact after this translocation, meaning there can be some missing part or they can contain an extra material.

1.5 Copy number variations

Copy number variations are type of the unbalanced structural variants. As the term says, they cover changes on DNA linked to change of number of copies and thus change of amount of genetic material. In this section we describe CNVs in more detail as soon as we are interested in copy number variants in this thesis.

In general, copy number variations represents a section of a genome that is copied multiple times, e.g. AAAC with copy number 4 results in AAACAAACAAACAAAC. The case when the section is present in genome in zero copies (meaning it is not in genome) also falls into the category of CNVs. This implies that deletions belong to CNVs as well. Both types of CNV are displayed in Figure 1.5. Copies of a certain

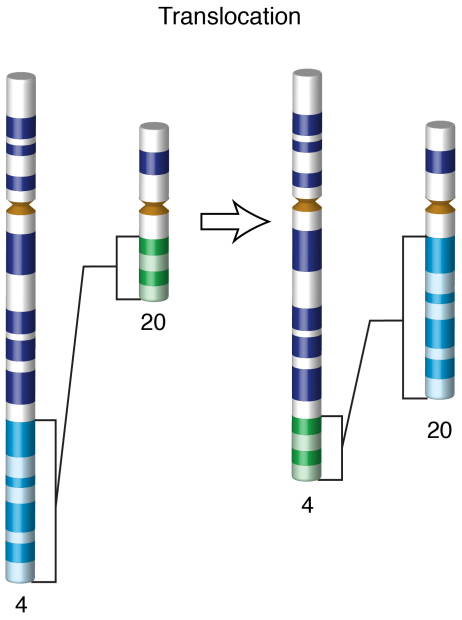

Figure 1.4: This figure show two chromosomes before translocation and after translocation. The light blue section from chromosome 4 and the green section from chromosome 20 have been switched.

region of DNA are almost identical. This phenomenon does not alter gene structure, if the whole gene is present in copied section, therefore the product of the gene remains unchanged. However, the amount of the gene product can vary from standard quantity.

Size of CNVs are variable as they can affect sections from 1kb to whole chromosomes. When the chromosome is affected by CNV, it is referred to as aneuploidy. The term monosomy represents deletions of a chromosome, while increased number of chromosomes is called polysomy. Infamous example of polysomy is a trisomy of chromosome 21, know as Down syndrome. Patient with this disorder has three complete copies of entire chromosome 21, what is termed full trisomy. In human genome may occur full trisomies of other chromosomes as well. However, the only known full monosomy in humans is monosomy of chromosome X. For this disorder, short stature, webbed neck and low-set ears are characteristic. Full monosomy of any other chromosome is lethal, but partial monosomies may occur.

Although CNVs are connected with syndromes and genetic malfunctions, they are equally related with evolution and positive selection. CNVs participate in formatting of genetic variability and thus they are creating opportunities for natural selection. In addition, CNVs create copies of genome section without damaging the function of genes in the original section. Therefore room for new positive mutations, which may innovate functions of the copied genes, is formed. New CNVs appear more frequently than other

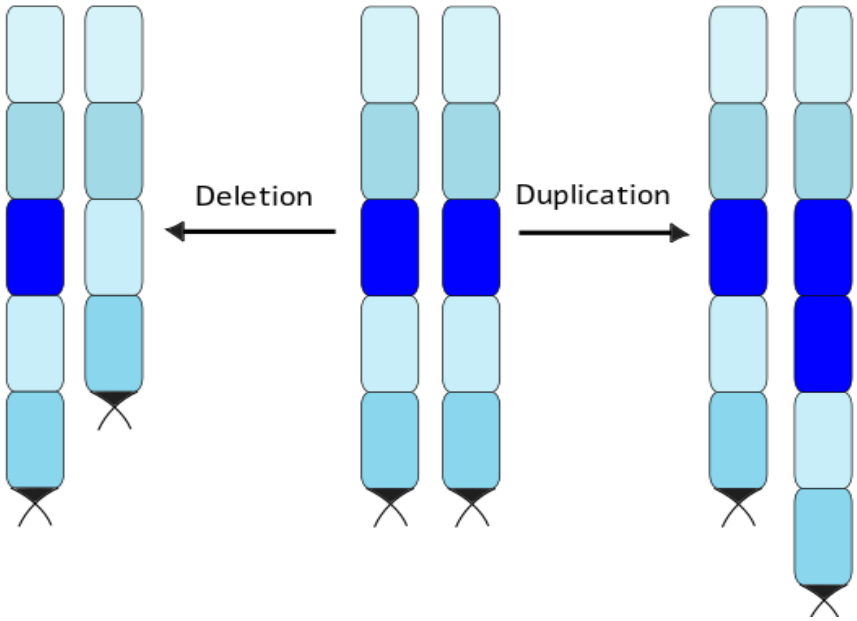

Figure 1.5: On this figure are shown both deletion and duplication of section of chromosome.

structural variants, which implies higher likelihood of positive mutation [17].

CNVs represent around 5-9 % of human genome. Distribution of this phenomenon is not even across the human genome. Higher occurrence of CNVs was discovered in regions located near centromere and telomeres [37]. Some CNVs have no serious effect on phenotype and these are commonly small sized and located in intergenic regions. On the other hand, pathogenic CNVs, which have negative phenotypic manifestation, mostly cover several important genes and are of larger size. However, establish rate of pathogenicity of all CNVs and ascertain their phenotypic manifestation still remains a difficult task [6].

1.5.1 Clinical interpretation of CNVs

In our thesis we are mainly interested in predicting possible impact of CNVs detected in genome of fetus. Various CNV detecting tools are used to determine particular type of disorder in the sequenced samples of the DNA of fetus. Next generation sequencing approach proved to be reliable for this type of structural variants detection [38].

If a specific duplication or deletion is found four major factors have to be considered [30]:

- **Parental inheritance** - Other members of family have to be tested as well. The same mutations may be discovered, which may simplify the prediction.

- **Databases** - Various CNVs are already recorded with description of phenotype. Databases such as Online Mendelian Inheritance in Man (OMIM), human genome browsers (UCSC, Ensembl) or DatabasE of Chromosomal Imbalance and Phenotype in Humans using Ensembl Resources (DECIPHER) may be reliable source of information.

- **CNV size** - Large deletions or duplications are more likely to affect essential segments of DNA. Although smaller CNVs should not be underestimated, the larger affected sequence, the more attention it requires.

- **Genomic content** - Mutations in coding regions may lead to more serious manifestations. Especially if affected genes are known to be associated with recessive or dominant diseases, in addition deletion of vital genes may be lethal.

1.6 Problem statement

As was mentioned above, mutations and structural variations in human genome can have negative impact on individual's health and overall phenotype. Even if the CNVs are detected in genome of a fetus, prediction of their potential manifestations yet remains a challenge.

The main aim of this thesis is to assess the impact of these genomic changes on the phenotype of individual. With this intention we inspect the possibility of using data about structural variations collected in publicly available databases.

Our objective is to create a model, which could assist in phenotype prediction of a tested patient. Inputs for this models will be coordinates of detected CNV such as start position and end position on a specific chromosome and for this input, our models will yield a set of possible characteristics expected to occur in resulting phenotype. As we are breaking new ground, to simplify our models, we consider only deletions of CNVs, as duplications would brought more, yet unwelcome complexity to our models.

In case our models would have acceptable success rate in its testing, it may be extended to application usable in clinical practice.

Chapter 2

Data sources and modules

In this chapter we introduce databases from which we have obtained our data. We describe data sources — some we describe in more detail. We subsequently briefly describe used programming language and libraries we utilized.

2.1 Human Phenotype Ontology

Human Phenotype Ontology (HPO) is a large ontology (or vocabulary) of phenotypes and phenotypic abnormalities that are related to gene mutations and diseases. The HPO database was created as most significant product of Monarch Initiative, a web platform focused to connect phenotypic and genotypic information across species. The HPO is aimed at supporting researches in field of bioinformatics and at helping with clinical diagnosis as well. In these days Human Phenotype Ontology cover over 13,000 terms and over 156,000 annotations to genetic diseases [23]. Structure of data in this database is reasonably organized. In our work we use benefits of its representation in considerable extent.

2.1.1 HPO term

Each term in Human Phenotype Ontology represents a clinical disorder. It could be a very general term, or on the other hand, the term could be very specific. For instance term *Abnormality of the skeletal system* is more general, whereas *Pituitary calcification* describes specific disorder. General terms are usually higher in the graph structure and represent parents of specialized and very specific terms. These terms, which symbolize specific genetic disorders and phenotypic abnormalities, are in the deepest nodes of directed acyclic graph (hereafter know as leaves) of ontology and thus they do not have children.

All terms have their unique identifier named HPO_ID and label or name called DB_name, for example term with label *Synotia* has ID *HP:0100663*. In addition, most

of Human Phenotype Ontology terms have a supplemental description or definition, which provides detailed characterization and more comprehensible explanation of the disorder or abnormality. As an example term *Synotia*, has a definition *A congenital malformation characterized by the union or approximation of the ears in front of the neck, often accompanied by the absence or defective development of the lower jaw.*

Human Phenotype Ontology database has came into existence by joining medical, clinical, genetic and bioinformatics resources. Therefore a significant number of terms and expressions with the identical meaning have lot of various names. Taking this into consideration, terms store their different synonymous names in annotation *synonyms*. For instance the term *Microtia* has various synonyms such as *Hypoplastic ears, Small ears, Hypoplasia of the external ear, Underdeveloped ears* and others.

Human Phenotype Ontology is divided into five sub-ontologies: *Phenotypic abnormality*, *Mode of Inheritance*, *Clinical modifier*, *Clinical course* and *Frequency* and each term is assigned to one of the sub-ontologies. Every sub-ontology adds extra information characterizing the term in more details.

Besides assigning to one of sub-ontologies, every term in HPO is described and characterized in detail by annotations. These annotations are statements that show association between particular HPO terms and specific genes or diseases. Not only specific terms have these annotations, but all their ancestors and all terms of database are precisely annotated. HPO annotations follow a strict format, which is composed of a certain number of columns. Each column stores specific information that may be required or optional. In our work, only two of them are important: HPO_ID, which is most cardinal field of every term in this ontology, and DB_name. DB_name is word appellation of the phenotypic abnormality.

2.1.2 Structure of Human Phenotype Ontology

HPO is represented by directed acyclic graph, what is similar to hierarchy, or maybe to tree representation. However, the main difference is that more specified term can have multiple parent terms with lesser specification. This property of directed acyclic graph can be shown in this example: the HPO term *Iris Coloboma* is child of the term *Abnormality iris morphology*, but is also child of the term *Coloboma*. Ontology without this property would not be so flexible and so comfortable to work with.

In addition, the relationship between two Human Phenotype Ontology terms is expressed by *is-a* edges. This connection means that one more specialized term is-a subclass of its parent, which is less specific and more general. For illustration, the Human Phenotype Ontology term named *Cardiovascular calcification* is-a (is a subclass of) term named *Abnormality of cardiovascular system morphology*. Is-a relationships are inherited from the lowest terms up to the root of Human Phenotype Ontology.

This characteristic of is-a relationship is called transitivity.

2.1.3 Formats of Human Phenotype Ontology

The Human Phenotype Ontology database was made and is available in two database formats. Each of them has its specifics and advantages just as disadvantages. Both formats are downloadable directly from the official web site of HPO [23].

- The OBO flat file format - OBO is an abbreviation for the title Open Biomedical and Biological Ontologies and it represents a language for creating ontologies. This format was invented to be as much as possibly comfortable and comprehensible for humans. As of today, it is widely used as a language for biological ontologies. There are also programming libraries specialized for work with this format of database.

- OWL - OWL means Web Ontology Language. It is a language primarily made for working and designing ontologies. The OWL version of Human Phenotype Ontology is a full version of this database, while the OBO version is simplified version of it. Therefore OWL version has some features, that are not present in the shortened OBO version, however as in our work there is no need for these features, OBO version will be sufficient for us.

2.2 ClinVar

ClinVar is a freely available database providing source of information for the public, maintained at the National Institutes of Health [25] [28]. It stores records of variations in human genome and their impact on phenotype. The archive contains any genomic variations of human genome. Records of any type or size are stored in database, as well as germline and somatic variants or records with various genomic location. Researchers, scientists, laboratories and others can submit variations, which they detected in their patients, with proper description and details regarding the diseases the patients manifest. After submission, the record is labeled by accession in SCV format, for example SCV000184036 [24]. If there are multiple submissions with equal combination of variation and phenotype, ClinVar assembles them together and labels them by accession in RCF format, as for example RCV000129276. If any conflicts in interpretation occur, they are reported.

Data from this database are necessary for our work and since it provides entries from clinical practice, our work does not need to rely on simulated or artificial data.

2.2.1 Database content

Content of ClinVar database consists of five main classes, which are: *submitter*, who provides data, detected *variation*, *phenotype* related to this variation, *evidence* of manifestation of variation, and *interpretation* of said evidence [24]. A record unit is specified by a unique union of three of them, which are submitter, variation and phenotype.

ClinVar database maintainers make effort to ensure strict structure of the content, which is at disposal for public. Some of details of the content format are explained below in description of classes.

- *Submitter* Submit their observations to ClinVar database may both individuals and organizations [26]. ClinVar stores information about all submission on its websites.

- *Variation* is major category of ClinVar database content [26]. The class variation connects two pieces of information about the genomic change. An information about the location of variation on genome and about the altered sequence as well. The category variation in ClinVar is represented as a set of genomic variations, however many of them contain only one genomic variation. The main function of variations in ClinVar database is to represent the relation between genomic variations and the human health as accurate as possible.

- *Phenotype* as category in ClinVar is represented by MedGen entry [24]. (MedGen is portal holding information on phenotypes related to medical genetics.) Although submitters are emboldened to use identifiers from other official databases, for example term ID from HPO, ClinVar accepts textual descriptions as well. If there is not any suitable MedGen record to substitute this textual description, new MedGen entry is created and used.

  As in the case of variation category, phenotype can consist of set of values in ClinVar as well. To describe overall phenotype caused by genetic alternation are used mostly sets. Single value in ClinVar phenotype category are used more likely as diagnostic terms.

- *Evidence* Function of evidence in database is to support interpretation of relation between phenotype and variation in submitted record [26]. Evidence may have various structure. It can be represented as textual description of obtaining the evidence. Or evidence can store the information, if there were any other variations observed together with submitted variation. Those evidence records with structure include names of variants or description of context, for example genetic testing, etc.

- *Interpretation* category explains the relation between submitted variation and phenotype [24]. Records in this category are provided to ClinVar database solely by submitters.

2.3 AnnotSV

AnnotSV is a program intended to annotate given structural variations [13]. In AnnotSV, significant information about functions of structural variations and clinically important information are connected. The main goal of AnnotSV is to provide high quality annotations, which may be used to detect potentially pathogenic structural variations. Beside this, the annotations may be useful to discover and separate variations with potential to be false positive, as well.

AnnotSV is accessible as a stand-alone program and after installation can be used as a command-line tool. It is designed to be easily executed on most of operating systems. However, on the website of AnnotSV, an interface for running AnnotSV online is available [13].

Similarly to ClinVar database, AnnotSV provides important valid data used in our work. Purpose of both, AnnotSV and ClinVar is described in a following chapter.

2.3.1 Annotation process

As an input, AnnotSV accepts files in *VCF* or *BED* format. VCF is an abbreviation for variant call format, which is a text file format for storing data of structural variations. VCF file consists of a header with information about body of file and the body as such. The body is divided into eight required columns and several optional columns. The BED format, which stores variations data as well, requires three columns with specified content and nine optional columns, however the order of optional columns is defined and when certain column is used, all previous columns have to be filled as well.

From the input file AnnotSV takes coordinates of variation. At first, the overlap between variation and annotation features is calculated and then gene names with connected annotations are reported. At the end, the output of the program consists of records, which store for each variation several entries. One annotation is stored for whole section of genome affected by variation and every gene from interval between the coordinates of variations is stored its own annotation [13]. The output file is in TSV format, which represents values separated by tabulator.

As platform for annotation process may be used reference genome GRCh38 or GRCh37, as the user defines. GRCh38 is the Genome Reference Consortium Human genome build 38 [15], the latest version of reference human genome, and it is built from genomes of several individuals. GRCh37 is a previous standard reference sequence,

which was as all previous reference sequences, from one individual [15].

2.4 Python and modules

Our partial aim was to develop a program, which would be able to obtain and display a subgraph of Human Phenotype Ontology graph containing HPO terms in nodes. After a research of available options regarding potential programming languages, we decided to use Python3 programming language as it is both easy-to-use with great community support and has all libraries we need.

First we tried to found libraries which worked with Human Phenotype Ontology directly. We managed to find two of them:

- *phenopy* - This Python package does work with Human Phenotype Ontology. More precisely, it is made to score similarity of phenotypes using their semantic similarity [22]. Semantic similarity is a metric which represent similarity between terms or documents. This similarity proceeds from resemblance between meanings of contents of terms or documents. It does not compare the lexicographic similarity.

  After a deeper research it became apparent that this library is not suitable for our needs, since it was specialized only to score similarity in various ways.

- *pyhpo* - Pyhpo library seemed to be appropriate to our thesis, however shortly before our installation this library became unavailable due to unknown reasons.

Finally we decided to create our programs using libraries, which are not focused directly on Human Phenotype Ontology, but they are specialized in graph manipulation.

2.4.1 obonet

Obonet is Python package designed for parsing ontologies in OBO format into a format, which package NetworkX can easily work with [18]. Obonet has implemented functions, which can read OBO-formated files from several types of input and subsequently transform it into *networx.MultiDiGraph* representation. For this use function *obonet.read_ obo()* is created. The input files can be loaded from open file, URL or from a path. In our project we have used *obonet 0.2.5* version.

Human Phenotype Ontology is available in OBO version. Obonet is the only Python package we have found, which handles with OBO formatted files. In addition it turns this files into more comfortable format in a way, that is very practical and easy for user. Therefore we chose this package in our work.

2.4.2 NetworkX

This Python package is very extensive and it gives its users wide possibilities to create, manipulate and study graphs and other networks [16]. NetworkX includes several classes for graphs, directed graphs and multigraphs and is able to convert them from and into various formats. Naturally, common graph functions are incorporated, such as function for the shortest path search, obtaining subgraphs etc.

In this project *NetworkX 2.4* version was used [29], since it provides broad range of functions. In addition, we can easily obtain NetworkX representation from OBO format of Human Phenotype Ontology using obonet package, which is described above. For the purpose of this thesis, class MultiDiGraph was used. This class represents and stores directed graphs. Directed graph is a suitable representation for Human Phenotype Ontology since this representation covers also directed acyclic graph, which is a structure of HPO itself.

However, in order to represent results, a reasonable visualization was needed as well. Even though networkX offers some basic visualization using matplotlib library (described below), other graph visualization tools are recommended. We decided to try the matplotlib library, because unlike other recommended tools, no graph conversion is needed.

2.4.3 matplotlib

Matplotlib is an comprehensive library for visualization of graphs [19]. It is compatible with different graphical interfaces from command line to web applications. Through this library NetworkX graphs can be easily visualized.

Matplotlib is popular among Python programming community especially for visualization of different graph plot types, such as pie plot, bar graph or histogram [21]. However, it is not efficient for displaying graphs with nodes and edges, which turned out to be problematic. Although its usage was seemingly practical, since no graph format conversion was necessary, visualization by this library did not meet our requirements. Therefore, we decided to select a package from tools recommended for NetworkX graphs.

2.4.4 PyGraphviz

After we did not succeed with visualization of NetworkX graph using matplotlib, we decided to use Graphviz library, or more precisely PyGraphviz. PyGraphviz provides a Python interface, which enables handling Graphviz library for displaying graphs [3]. It offers ability to create graph and comfortably set colors or shape of nodes, their labels and others.

Although exporting graph from NetworkX MultiDiGraph format to AGraph is required, programming interface provide by PyGraphviz is comparable with NetworkX. That is the reason we chose this Python package.

Graphviz

As the name indicates, this library is focused on visual representation of graphs and networks. This way of displaying information is very useful and Graphviz library affords many practical features for editing and developing graphs [9] [12]. It enables users to set font, many various shapes of nodes and also edges, node layouts and even more.

Layout programs of this library can make the visual representation in formats such as images and SVG useful for web pages. To create them, graph described in simple textual form is required. This can be done manually, using graphical editor, or simply in text file. However, in most cases are graphs obtained from external sources, such as being converted from other formats.

2.4.5 FuzzyWuzzy

In this thesis it was necessary to match HPO terms with textual description of phenotype. However, not all of this descriptions were equal to term names. Therefore the FuzzyWuzzy package was used.

FuzzyWuzzy is Python package, which is designed to compare strings according to their lexicographical similarity [1]. It provides several methods for comparing two strings and also for comparing one string to list of strings.

Since FuzzyWuzzy uses Levenshtein distance for its computations, installation of python-Levenshtein is demanded.

Levenshtein distance is string metric, which defines the difference between two string as the smallest possible number of changes on single character needed to make the strings equal. These changes include insertions, deletions and substitutions of a single character.

python-Levenshtein

This module calculates Levenshtein distance and several types of string similarities [2]. However, for FuzzyWuzzy is sufficient just to have this module installed. Therefore we do not use python-Levensthein directly in our work.

2.4.6 colour

This library is used to convert between many various formats of color representation, such as RGB, HSL, six digit hexadecimal and others [36]. It also offers very simple and

intuitive way to create colors and handle them. In our work we need to automatically assigns colors to objects, thus this library was used.

Chapter 3

Data preparation

Data for our work were obtained from the ClinVar database, which is described in Chapter 2 [25]. ClinVar database provides its data in XML format, however they were converted to TSV format, since this format is more suitable for our needs.

3.1 Reducing surplus data

The table we obtained stores in every row several information about one CNV. In this data a huge amount of information in many columns was stored. However, the reduction was necessary, due to excess of information, that were not useful for our work. As was mentioned, we consider only deletions of CNVs. Thus solely entries with *copy number loss* in column *type* were selected from all the entries in this data.

Using linux command line command we removed surplus information and maintained only six columns from the original table - *RCV accession*, *chromosome*, where the CNV occurred, *starting position* in this chromosome and *ending position*, *disease finding* and the *order number*. The order number was retained to be used in case it would be necessary to look up the particular CNV in the original table.

Each CNV is labeled by RCV accession, which was assigned to it by ClinVar database. This accession is used as an ID for particular CNV in this work, although the combination of chromosome, start position and stop position is also unique for each CNV. However, in our data, there were some entries with missing records in this fields that we had to ommit.

3.2 Subgraphs for phenotypes

One of the most significant column in our data is *disease finding*. In this column is stored information of phenotypes (some CNVs have more then one phenotype recorded), which are linked to this CNV and are recorded to ClinVar database. On the basis of

this phenotype information we created subgraph of the HPO graph for each CNV in our data. New column representing this subgraph was added. It stores IDs of each HPO term, which occurs in particular graph.

In the table resulting from previous adjustments, there were some entries with string value "not provided" recorded in this column that we had to remove.

Our intent was to each phenotype of one CNV find corresponding term and create its subgraph. After that merge the subgraphs and then the terms from this subgraph write down to the table. However, it became apparent that its not possible to search in the terms using directly records from *disease finding* column, since most of them do not equal with any term names from the HPO database. But there were obvious similarities between this records and the corresponding terms. This corresponding terms we found manually, to find out how to match the records with the HPO terms.

To clarify this may be used this example with artificial values: In our data, there is a record in *diesease finding* column with string value "Coloboma of iris", but there is no term in HPO with this name. In the HPO database is only term *HP:0000612* with name *Iris coloboma*.We considered the possibility to match each individual word from *disease finding* column with each word from the term name, and value each term with some matching score. But this method appeared to be too complex for further implementation.

3.2.1 FuzzyWuzzy Python package

We decided to use a python package to find the most matching terms. The package FuzzyWuzzy has several useful methods. Especially the method process.extractOne() was effective in solving our situation. This method gets as arguments one string and list of string to compare, and it choose one string from the list which has the highest ratio of match. We used this method giving it a value from *disease finding* column and a list of all names of terms from the HPO database. We also manually verified if this method matches the terms properly by choosing several random *disease finding* records and inspecting the terms, which were assigned to them.

A problem occurred with record "Developmental delay AND/OR...", which has no matching term in the HPO database. We tried to replace it using string value "Developmental delay", however this value was still very general and there was not any appropriate term for it, only eight similar terms, but each of them was more specific to match them for certain. Although this record was very frequent (it comprised almost 31% of all recorded phenotypes) in our table, it has very weak value in describing phenotype. Thus we decided to remove it from our data.

3.2.2 Error in FuzzyWuzzy

Later, during our work we realized that one *disease finding* value is not assigned to corresponding subgraph, i.e. term. It was a record with string *Keratocystic odontogenic tumors of jaws* in *disease finding* column and the term *HP:0000002*, which was assigned to it represented phenotype *Abnormality of body height*. This two phenotypes do not correspond obviously. Appropriate term from the HPO found manually is the term *HP:0010603* representing *Odontogenic keratocysts of the jaw*.

We made several experiments to find were the error appeared. Using methods of FuzzyWuzzy package we noticed, that method fuzz.ratio(), which gets two string and returns ratio of their match, returns higher value when it gets *Keratocystic odontogenic tumors of jaws* with *Abnormality of body height*, than when the *Odontogenic keratocysts of the jaw* and *Keratocystic odontogenic tumors of jaws* are used as arguments.

Probably there is some mistake in this package and we will report this finding to authors of this package.

After this discovery, we manually inspected several hundreds of our records, however no more mismatches were found. All the mistakes in our data connected to record of *Keratocystic odontogenic tumors of jaws* were fixed using specific condition in the program.

3.3 Assigning genes to CNVs

Next information we needed to add into our table was information comprising the genes affected by CNVs. We used AnnotSV tool, which for every sequence (defined by start and end position) on a particular chromosome determines all genes, which occurs in this section of genome and thus are hit by the structural variation located in this position.

We took all the starting and ending coordinates together with corresponding chromosomes from our table, and we stored them in a file in BED format. The fourth column was added to the three original columns containing extra information that all the entries were representing deletions.

Data we obtained from AnnotSV were more extensive, than we needed. Entries in this data are of two types. First of them has record *split* in the column *AnnotSV type* and second is connected to record *full* in the same column. This values refer to the fact, that the first type of entries represent records about only one of gene affected by CNV and for each CNV all the affected genes are stored in this way. The second type represents all genes connected to particular CNV stored in one record. For our work the *full* record is relevant.

Finally the genes were added to our table storing CNVs and information about

phenotype in form of new column. CNVs from phenotype table and gene table were identified and connected on the basis of unique combination of chromosome and coordinates.

3.4 Gene-phenotype relation

Although we had interconnect genes to phenotypes trough their connection to specific CNV, relationship between genes and phenotypes, which apply to them is described separately. File characterizing this connection is publicly available via HPO official website. This file is in TXT format however we converted it to TSV format.

This file contains genes stored individually in rows and each gene is linked to several phenotypes described as a specific HPO terms and also IDs of this terms. Other information from this table are insignificant for us, thus we extracted only the tree mentioned columns - the gene, related phenotype and the ID of term. Thereafter we created a subgraph of HPO for each term in this table as well, and added this information in form of set of terms into the table as an extra column. Finally we sorted the table according to names of genes to simplify using it in further work. Further in the text we refer to this data as to a table of genes, or gene table.

3.5 Explicit term

During following steps, the need of having **explicit terms** stored individually in our data appeared. Explicit term is the term, which name was matched with *disease finding* directly . We added a new column storing this information and filled it with explicit terms for each CNV, again using FuzzyWuzzy.

Chapter 4

Visualization of HPO subgraph

As was mentioned in Chapter 2, we made an effort to create a model which would made a prediction of resulting phenotype for given CNV coordinates. This model will yield a set of possible manifestations of entered CNV. These will be represented as subgraph of HPO graph, in which the possible phenotypes will be present in form of corresponding HPO terms.

4.1 Subgraph of the HPO graph

Since the HPO graph is directed and acyclic, the subgraph of a specific term stands for a graph of all the terms accessible from this specific terms trough directed edges. Hereafter in connection with graphs, the word *term* is used synonymously for the *node*, in this thesis.

The edges in the HPO graph represent "is-a" relationship of two connected terms, so they are oriented from leaves to root of the graph, what is the term *All* with ID *HP:0000001*. This implies that the subgraph of the specific term includes parents of the term and all the ancestors of it to the root term.

As a first step we wanted to write a program, which would take as an input name or ID of a specific HPO term and obtain a subgraph of said term from the entire HPO graph. Since every term has a unique name and a unique ID, to identify a term, any of both is sufficient. Although terms are reachable in the HPO graph on the basis of ID only, Python dictionary connecting names and ID was our solution.

4.1.1 Packages for the HPO manipulation

We needed to be able to read the HPO database and transform it into graph representation, so that we could easily search and manipulate with this graph. We tried to find Python modules, which work directly with the HPO graph and which could satisfy our demands. As we have mentioned before, we found two python packages (phenopy and

pyhpo) working with HPO, but unfortunately they do not meet our needs. They are described in more detail in Chapter 2.

4.1.2 Reading the HPO

After the failure with the two modules above we attempted to find another way to work with the HPO in graph representation. The HPO database is freely available in two formats, the OBO and the OWL. OBO format of HPO database is more simple, as the OWL format contains some extra features, that we do not need for our work. Thus the obonet package, which can handle with OBO-formated databases was very practical choice. In addition, obonet function obonet.read_obo() transforms this format of database to a representation of NetworkX class MultiDiGraph.

4.1.3 Graph manipulations

NetworkX is very practical package and it is highly suitable for our intention. We appreciated especially functions networkx.algorithms.dag.ancestors(), networkx.algorithms.dag.descendants() and networkx.Graph.subgraph().

First two of them get as an argument a node, which ancestors/descendants we request and the graph, where the node is located. They return a set of nodes, which are reachable from the given node. While finding the reachable nodes, this function moves through the graph in the direction of the edges or in opposite direction. The direction of this movement depends on what set of nodes is called, whether the ancestors or the descendants.

In general, edges in directed graphs are oriented from root down to leaves. Thus the set of ancestors usually refers to set of nodes reachable in reverse direction to orientation of edges. Due to the fact, that edges in HPO are oriented from leaves to the root, we had to use function networkx.algorithms.dag.descendants(), to get the ancestors of the node.

Function networkx.Graph.subgraph() takes list of nodes as an input and returns a MultiDiGraph with the nodes from the list only as well as edges connecting only these nodes. Using these functions we managed to accomplish the first step.

4.1.4 Vizualization using matplotlib

As soon as we were able to extract desired subgraph from the HPO graph, we tried to visualize it. NetworX offered some basic visualization using the package matplotlib. Although using other packages was recommended, visualization by matplotlib does not require conversion of graph from NetworkX format to another, thus the matplotlib was used.

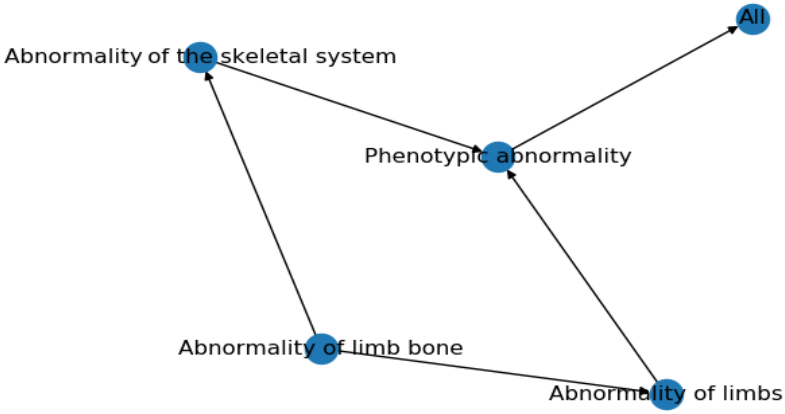

Figure 4.1: HPO term *Abnormality of limb bone* with ID *HP:0040068* visualized using matplotlib package.

Example in the Figure 4.1 displays how the visualized graphs looked like. This visualization is not perfect, however it appeared acceptable for our uses.

4.2 Score and merging of subgraphs

Next important feature we added to our work was **impact score**. Impact score is numeric value, which is assigned to every term in subgraph. Calculation and usage of this value for each term is explained and specified below. Function of impact score is to determine the expected impact of a given term on the overall phenotype. It enables comparison of the importance of impact of the terms on the phenotype.

Impact score is essential in **merging** the subgraphs as well. In this thesis a graph, which results from the process of merging two subgraphs of HPO is also a subgraph of HPO, which includes all the terms from both subgraphs. The terms, which occurred in both subgraphs are in merged graph present in only one copy and its impact score is summed from the terms in both subgraphs. The impact score of the terms, which occurred in only one of the two subgraphs stays unchanged.

To illustrate the merging of two subgraphs the situation displayed in the Figures 4.2 and 4.3 can be used. The values in labels of nodes represent the impact score. In this case artificial values of impact score were used, to simplify the example. The figure 4.2 shows two subgraphs of the HPO before merging process. On the figure 4.3 is displayed subgraph after merging. It is apparent, that impact score of terms *All* and *Phenotypic abnormality* results from sum of values from the original subgraphs.

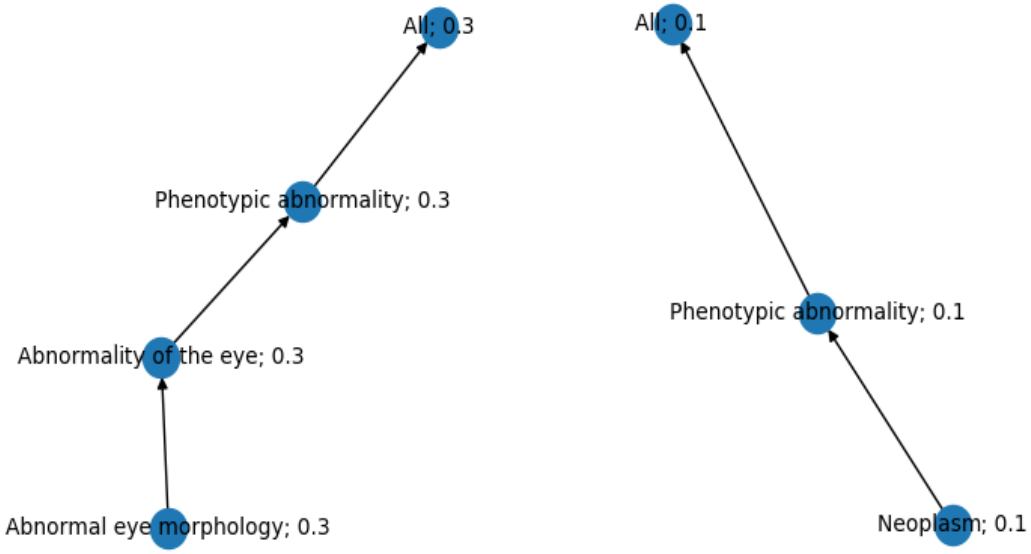

Figure 4.2: On the left side of this figure is shown subgraph of the term *Abnormal eye morphology* with ID *HP:0012372*, on the right side is *Neoplasm* with ID *HP:0002664*. Both of them are visualized using matplotlib.

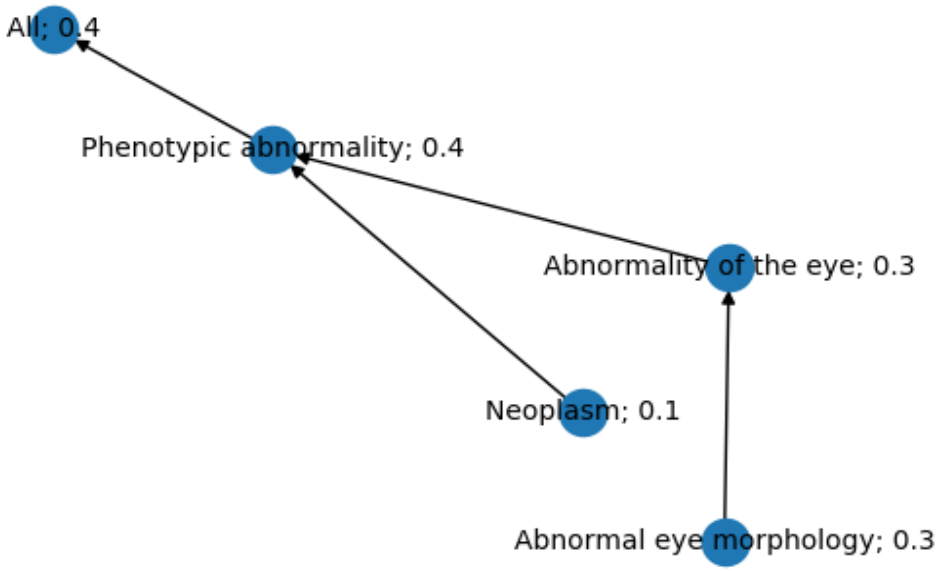

Figure 4.3: This figure displays subgraph resulting from merging the two subgraphs from the example on figure 4.2.

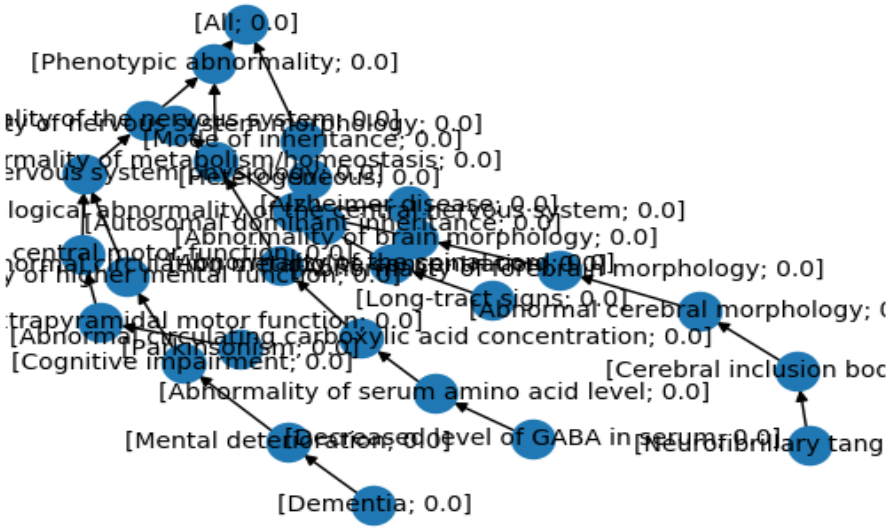

Figure 4.4: This figure shows how unclear is our visualization in this phase of our work.

4.3 Visualization improvement

We prepared all the necessary data and subsequently we attempted to create and visualize graphs directly from the data.

4.3.1 Fisrt visualization of HPO subgraph from gene

The first data which we visualized were genes from the our gene table. We wrote program, which for input gene finds all occurrences of this gene in our gene table. For each occurrence, the subgraph of related phenotype is taken and merged to the unified subgraph. As a result from this program we obtained graph representing all the phenotypes of the particular gene. Unfortunately the visualization was highly unsatisfactory. In this stage we were still using the matplotlib visualization, which appeared to be not suitable for larger graphs of this type.

To clarify this by example we use Figure 4.4 where is visualized gene A2M using matplotlib visualization. Graph of this gene is one of the smallest and most simple graphs among genes in our data, however the visualization is still very disorganized.

In this stage of our work the impactScore is not defined yet, thus artificial zero values are used till now.

4.3.2 Solution using PyGraphviz library

After the visualization of NetworkX graph using MatPlotLib rendered unsatisfactory, we were constrained to substitute it with another library. As was mentioned in Chapter **??** the library PyGraphviz offers programming interface similar to NetworkX package albeit conversion of NetworkX graph to PyGraphviz graph format is needed. However, this conversion is very uncomplicated since NetworkX provides a method networkx.to_agraph(G) designed directly to convert given graph G in some NetworkX graph format to an AGraph format of Pygraphviz.

4.3.3 Error in PyGraphviz

The program was adjusted (including graph conversion) for using PyGraphviz, however the program was able to visualize only a small portion of genes from our data. If the other genes were entered, individually or in pairs, our program ended up with error report. We tried also enter two copies of the same gene, which do not cause error inputted in one copy, as in case of merging of two graphs and the error occurred even with this input. We were however able to locate and adjust our data to accommodate this error.

In the OBO formatted ontologies, some of the terms do have in their annotations a string \" (backslash and quotation marks, without space) meaning as an escape for " character. For example in the term *HP:0001166* named *Arachnodactyly* is definition *"Abnormally long and slender fingers (\"spider fingers\")."* Pygraphviz however did not escape these quotation marks as the backslash defines and instead it ends the string, making the rest of the string a command in the underlying C library that naturally yields a syntax error. Bearing in mind the tenets of Python programming language, we believe that a PyGraphviz, as a wrapper to a Graphviz library (written in C), should be able to deal with a quotation marks, thus we informed authors of the PyGraphviz package about this deficiency.

To be able to work with PyGraphviz and the HPO simultaneously, we modified our version of the HPO simply removing the \" string in all its occurrences. Since this string is used only in descriptions and definitions of the terms, its removal do not have any negative effect on the function of the database.

Thereafter was our program functional again and we were able to visualize subgraph of all phenotypes related to particular gene. In the Figure 4.5 is visualized gene A2M - the same gene as on the Figure 4.4. The structure of the subgraph of HPO is obvious and labels do not overlap each other. Improvement compered to previous visualization is significant.

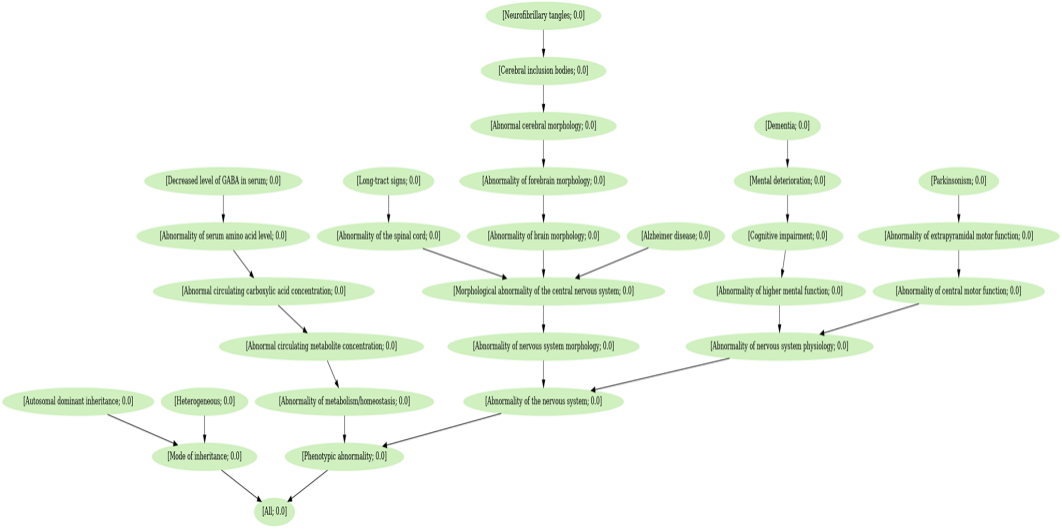

Figure 4.5: This figure represents example of subgraph of gene visualized using PyGraphviz.

4.3.4 Explicit terms in graph

After resolution of problem with PyGraphviz we intended to improve the visualization by highlighting the explicit term. Visualized graph with highlighted explicit terms is displayed in the Figure 4.6.

This faze of our work was also the moment, when was needed to add the explicit term into our data as well, so the gene table were extended into its final form.

We wrote two programs for visualization of subgraphs based on the data from this table. These programs get RCV accession of particular CNV, which we were using for identifying the CNVs. The first of the programs creates graph using subgraph of phenotypes related directly to particular CNV and stored in the table. Using only phenotype related to one given CNV, usually only small graphs were arising. One of them is displayed in the Figure 4.7. In connection with this CNV was noted down only one phenotype.

The second program takes, for the given CNV, list of related genes from the record in our CNV table and then for each gene creates subgraph on the basis of records from our gene table that links genes to phenotypes. The process of creating subgraph using genes is similar as the process the program from section above done, but this is done for more then two genes. Graphs obtained from this program were larger and more extensive, but still similar to Figure 4.6.

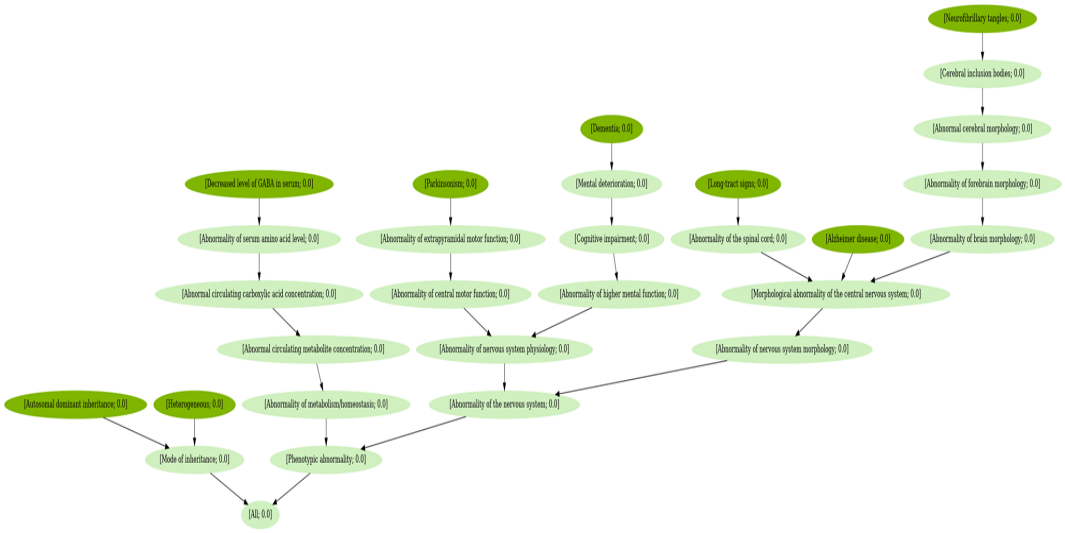

Figure 4.6: The graph on this figure represents subgraph of the gene A2M.

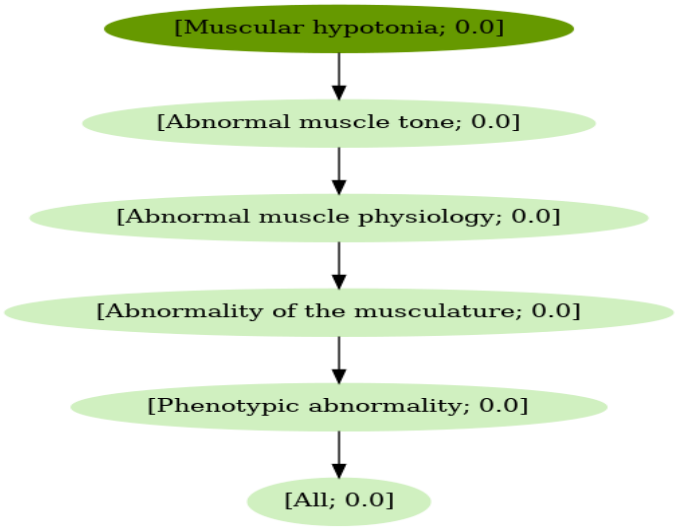

Figure 4.7: The graph created on the basis of phenotype directly connected to given CNV.

Chapter 5

Prediction models and results

Two visualization programs mentioned in previous Chapter **??** were extended and improved to become prediction models. We changed the input format from RCV accession to three of information, which determines the particular CNV equally as the RCV. New inputs — start position, end position and the chromosome — enables us to simulate possible real use of our work. The three values used as input comprise the information about deletion obtained through DNA sequencing.

5.1 Model based on CNV overlaps

Our first model is based on the assembly of the subgraphs of all **overlapping** CNVs from our data table.

Overlapping CNV is the CNV that occurred on the same chromosome as examined CNV and which does have certain number of bases common with entered CNV, e.g. a CNV defined by starting position on base 10, ending position on base 15 and chromosome 2. and a CNV on chromosome 2 with starting position 8 and ending position 12. These CNVs are overlapped since the bases 10, 11 and 12 were affected by both of CNVs.

The inputted values are coordinates of CNVs from our data as well. Naturally, when going through table, our model skips the record of the CNV, which was entered as an input. For this purpose the fourth information was inputed with CNV coordinates — the RCV accesion, which was used only to recognize particular CNV in our data.

Statistic of CNV overlaps

In this statistic we inspected the number of CNVs which do overlap. More preciously, for each CNV in our data we count the number of CNVs which do have any overlap with this specific CNV. As a result we found out that CNVs are overlapped by 61,5 of CNVs in average, while the highest number of overlaps of one CNV was 1323. Of

course there were also CNVs which are not overlapped by any other CNV, in our data. But there are only two CNVs of this kind in our data.

We also count average overlap ratio for each CNV. Overlap ratio of CNV is the ratio of number of overlapped bases and the length of the CNV from the data. Average overlap ratio is the average value calculated from all the overlaps which are related to this CNV.

The calculations showed that the maximum average overlap ratio ascertained in our data was 1. This means that the CNV was overlapped by the full length of the CNVs, which it shares bases. Unsurprisingly, in our data occurred also CNVs with zero average overlap ratio, but in average, considering average through whole data this value is 0,3.

Impact score in CNV overlap model

Until this stage of our work we were using artificial zero values as a score, however we finally defined how would be the impact score determined. The score for each term in this model is calculated as follows. All the terms from the subgraph of particular CNV are added to arising graph with score with the same value. This value represents ratio of overlapped bases to the number of bases of whole the particular CNV. Impact score of terms which already are in the graph is summed as was described below.

The score in resulting subgraph predicts the expected impact of this CNV on phenotype of individual.

5.2 Model based on affected genes

The second model gets chromosome and coordinates of CNV as an input, finds the related genes in our CNV data and creates the graph of phenotypes using the phenotypes related to genes from the table storing relation between genes and phenotypes.

Despite we use stored information, which genes were affected by CNV, usage of this model is not diffrent from possible usage in real life. In real life the information about affected genes can be obtained from the AnnotSV for each specific CNV determined by chromosome and coordinates.

5.2.1 Score in gene model

This model was extended by the impact score as well. Since in this case there is no overlap, the impacts score is determined in a simpler way. To each term from subgraph of phenotype related to specific gene is assigned score with value 1, as we expect that every gene is contributing with the same degree. Again, when the subgraphs are assembled, the score of terms, which occur more then once, are summed.

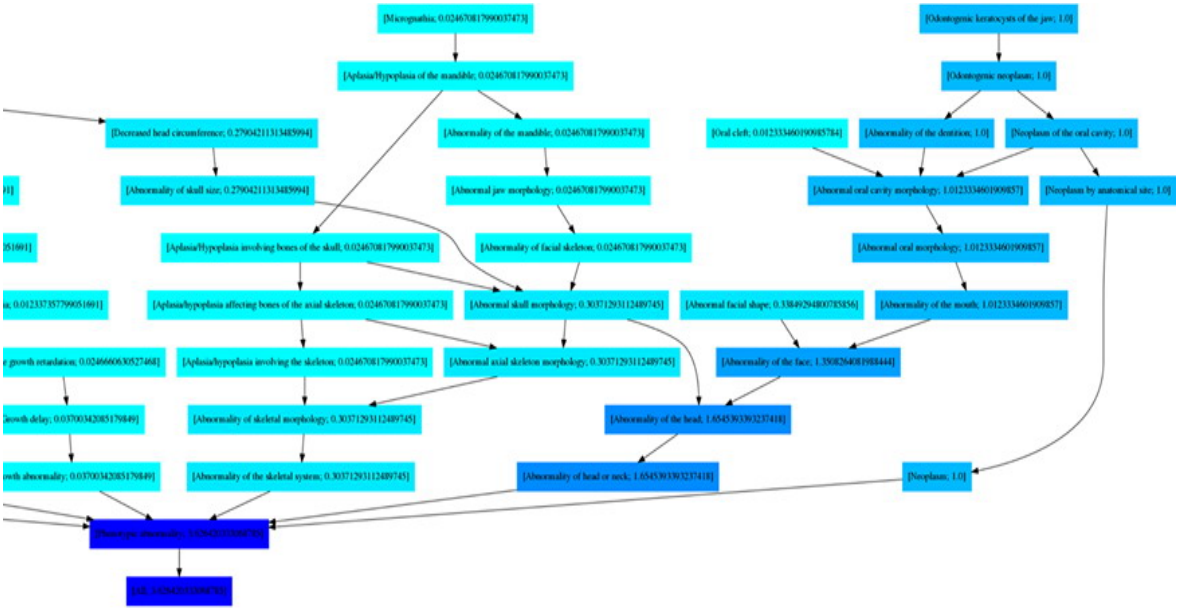

Figure 5.1: Displayed part of graph shows an example of visualization of different impact score values among the terms.

5.3 Visualization of the models using score

In visualization of graphs created by our models are different values of impact score represented by different shades of color. This shades from specific interval are assigned to terms according to their score. More preciously, the shade is determined by the ratio of the score of particular term to the highest score in the visualized graph. To set this the python package Colour was used.

Example of resulting graph is shown in the Figure 5.1. This figure shows part of graph obtained from the first model (CNVs overlap model). The difference between shades of terms are obvious. The darker terms on the right side of Figure refer to higher impact score. Naturally, the darkest shade, i.e. the. highest score, has the term *All* in the bottom of the Figure, since it is the root of HPO graph and thus is present in subgraphs of all the CNVs.

Visualization of the graph from the gene model is technically equivalent to the visualization of the CNV model, however graphs from gene model are usually much more extensive. Due to this fact visualization of gene model graph was not suitable to be exampled by figure.

Unsuitable term

During our tests on visualization we noticed that in almost every graph resulting from our CNVs overlap model the term *Global developmental delay* appears, representing a leaf with a significantly high impact score.

After we inspected our CNV table, we found out that from approximately 23 thousands of terms, almost six thousands (more than 25%) is comprised by the term *Global developmental delay*. Since this term is not very specific nor informative, we decided to remove it from our data as well as the term *Developmental delay* in previous stages of this work.

5.4 The third model

Besides the two mentioned model, we also created third version of prediction model. This complex model arose as connection of graphs from both previous models. It creates the graph using overlaps as well as genes for given chromosome and coordinates of CNV.

Since the impact scores in the two models are based on different principles, it is not possible to simply sum them together, when the same term occurs. To solve this problem the α value was defined.

α is a number, which determines the rate in which the two models contribute into resulting impact score. The value of score of the terms from each of the two models is modified according to the α. For example, if the α value is 0,3 the score values of the terms from the CNV model are multiplied by 0,3 and the gene model terms are multiplied by (1 - 0,3).

5.4.1 Appropriate value of α

Thereafter the α was added to the third model, the question, which value is appropriate for it, occurred. We decided to estimate the value of α, which would provide the highest possible accuracy of this model.

In order to estimate the value of α we added a support value M, which is related to the dept (the shortest path from term to the root) of the particular explicit term in the graph. Value M in the specific depth is than defined as a ratio of the number of terms with higher score than the score of explicit term in this specific depth to all the terms in this depth.

We tested ten various values from 0,1 to 1.0 with a step of 0,1 for the the α and we took approximately one fourth of the CNV data as data for calculation of α. For each of this values of α, we calculated an average value of M for each CNV in data. Thereafter we had the ten average M values for each CNV, one average M value for one value of α. From these α values were selected those with minimum average M value. The value of α, which has the minimum average M value in most of the cases, was proclaimed to be the most appropriate for the model.

The resulting alpha value was 1.0. Which means that the highest accuracy, the

Percentage of CNV	every term	at least one term	none term
CNV overlap model	37,2%	39,4%	60,6%
Gene model	17,6%	18,6%	81,4%
Complex model	42,0%	44,1%	55,9%

Table 5.1: Percentage value of CNVs with their covered explicit terms.

contribution of the gene model should not be taken into consideration. This fact is discussed in more detail below.

5.5 Statistics and results

To see to which extent are our models able to find the proper terms for given CNV we did some basis statistic.

Occurrence of explicit terms

As a first step, we compered the clinically observed phenotypes, which are recorded in our CNV data, with the phenotypes predicted by our model. For each CNV from our data (approximately eighteen thousands of entries) we count if the explicit terms of the particular CNV are present in prediction or not. In this statistic we did not consider the score of the terms.

In prediction made by the CNV model all explicit terms of CNV were included in 37,2% of cases. In 39,4 % of cases was included at least one of explicit terms and in 60,6% of cases were no explicit terms included in predicted graph.

Prediction made by the gene model has noticeably worse results. In 81,4% of cases no explicit term was found in predicted graph. All the explicit terms were included in 17,6% of cases and at least one explicit term in 18,6%.

The best result in this statistic reached the complex model. In 44,1% of cases at least one explicit term was found. All of them were present in 42,0% of cases and in 55,9% no explicit term was included. This result of complex model were obtained with α set on value 0,5.

Summarization of these values is presented in Table 5.1 In the columns are numbers of covered explicit terms.

We suppose that the more extensive data with higher amount of records of CNV–phenotype relationship would significantly increase the percentage of covered explicit terms.

The statistic for α with value 1,0 should be done as well, however the results would be identical with the results of the CNV model. This is implied by fact that complex

Alpha value	**0,1**	**0,3**	**0,5**	**0,8**	**1,0**
Maximum percentage	43,68%	33,22%	26,44%	18,34%	13,53%
Minimum percentage	0,70%	0,70%	0,70%	0,70%	0,70%
Average percentage	25,50%	21,73%	19,09%	15,33%	11,12%

Table 5.2: The table shows percentage relation between increasing α value and the values of average M.

model with α set on 1,0 do not consider the impact of genes, since their score is multiplied by zero.

Although the complex model with α value 0,5 has significantly higher occurrence of explicit terms in its graph, the most suitable α was calculated to be value 1,0. This is due to high extensiveness of the graphs from gene model. The graph from gene model contains large amount of terms, thus it is likely to increase the number of explicit terms occurred in it. However, simultaneously it increases the number of the terms with higher impact score then the explicit term. This is equal for the whole graph and for particular depths in this graph as well. Thus, higher number of terms in graph increases the M value as well and it cause the lower accuracy of model with α, which assigns higher score to terms from gene model.

Demonstration of this fact is represented by the following statistic.

M value statistic

We did the statistic for the M value as well. We inspected what percentage of terms in equal depth with specific explicit term of particular CNV does have higher score then this explicit term. The examination, how is this value influence by changes of α, was done.

The best results were obtained for α 1,0. The average M value for CNV in our data is 0,11, which means that there are 11% terms with higher score than the score of explicit term, in the specific depth of graph. The maximum M value was 0,13 and the best recorded percentage was 0,6%.

The attached Table 5.2 displays how is the average percentage connected to α. It is apparent that with increasing α the average percentage decreases. The same applies to maximum percentage. In our data CNV with no related gene occur as well. The minimum percentage is probably percentage of one of them and the changing weight of score of genes did not influence it.

-

Conclusion

In human genome various types of structural variants can occur. Although some of them are benign, others can have negative consequences on human phenotype, such as different syndromes or genetic disorders. Therefore it is important to correctly detect them and predict their possible impact on health of individual.

We implemented three prediction models based on different approaches. We discovered that prediction based solely on affected genes is too non-specific. This is due to the fact, that individual gene is often related to a great number of phenotypes.

To continue, the approach based on CNV overlaps yielded more satisfying results. In future work we would like to elaborate their full potential. CNV model was able to cover all clinically observed phenotypes of particular CNV in 37,2% of cases. The main factor that influenced the final percentage is the shortage of valid data. Our statistic proved that on average only 11% of terms in specific depth of the graph has higher impact score than the explicit term.

Originally we aimed to improve the prediction models, however our work was highly complicated by errors in used packages. Especially an error in PyGraphviz was of high significance, since this package is widely used. However, we were able to locate the source of both problems and subsequently adjust our data to omit these errors. We informed the authors about these inconveniences.

We assume that more extensive data with higher number of properly recorded CNVs with related phenotypes would bring relevant improvement. In future work research for additional valid data resources can result in noticeable progress.

We propose further improvement. Using only the subontology of the HPO named *Phenotypic abnormality* would reduce the number of terms in predicted graph, which could result to decrease of the terms with higher score than the explicit terms.

This work met our expectations and we believe that it can serve as foundation for future projects and researches.

Bibliography

[1] Adam Cohen. Fuzzywuzzy, last visited: 20 May, 2020. https://github.com/seatgeek/fuzzywuzzy.

[2] Antti Haapala. python-levenshtein, last visited: 20 May, 2020. https://github.com/ztane/python-Levenshtein.

[3] Aric Hagberg, Dan Schult, Manos Renieris. Pygraphviz, last visited: 23 May, 2020. http://pygraphviz.github.io/.

[4] Adam Auton, Gonçalo R. Abecasis, et al. A global reference for human genetic variation. *Nature*, 526:68–74, 2015.

[5] Erman Ayday, Emiliano De Cristofaro, Jean-Pierre Hubaux, and Gene Tsudik. The chills and thrills of whole genome sequencing. *Computer*, 06 2013.

[6] C. P. Canales and K. Walz. *Cellular and Animal Models in Human Genomics Research.* Academic Press, 2019.

[7] Nature Education. mutation, last visited: 26 April, 2020. https://www.nature.com/scitable/definition/mutation-8/.

[8] EMBL-EBI. Types of genetic variation, last visited: 2 May, 2020. https://www.ebi.ac.uk/training/online/course/human-genetic-variation-i-introduction-2019/what-genetic-variation/types-genetic-variation.

[9] Emden Gansner, John Ellson. Graphviz - graph visualization software, last visited: 23 May, 2020. https://www.graphviz.org/.

[10] Geòrgia Escaramís, Elisa Docampo, and Raquel Rabionet. A decade of structural variants: Description, history and methods to detect structural variation. *Briefings in functional genomics*, 14, 04 2015.

[11] National Center for Advancing Translational Science. Achondroplasia, last visited: 1 May, 2020. https://rarediseases.info.nih.gov/diseases/8173/achondroplasia.

[12] Emden R. Gansner and Stephen C. North. An open graph visualization system and its applications to software engineering. *SOFTWARE - PRACTICE AND EXPERIENCE*, 30(11):1203–1233, 2000.

[13] Veronique Geoffroy, Yvan Herenger, Kress Arnaud, et al. AnnotSV: an integrated tool for structural variations annotation. *Bioinformatics*, 34:3572–3574, 2018.

[14] AJF Griffiths, J H Miller, DT Suzuki, et al. *An Introduction to Genetic Analysis*. W. H. Freeman, 2000. Available from: `https://www.ncbi.nlm.nih.gov/books/NBK22042/`.

[15] Yan Guo, Yulin Dai, Yu Hui, et al. Improvements and impacts of GRCh38 human reference on high throughput sequencing data analysis. *Genomics*, 2:83–90, 2017.

[16] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[17] P. Hastings, J. Lupski, S. Rosenberg, et al. Mechanisms of change in gene copy number. *Nature Reviews Genetics*, 10:551–564, 2009.

[18] Daniel Himmelstein. obonet: load obo-formatted ontologies into networkx, last visited: 22 May, 2020. `https://github.com/dhimmel/obonet`.

[19] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science Engineering*, 9(3):90–95, 2007.

[20] National Human Genome Research Institute. Translocation, last visited: 24 April, 2020. `https://www.genome.gov/genetics-glossary/Translocation`.

[21] John Hunter, Darren Dale, Eric Firing. Matplotlib, last visited: 20 May, 2020. `https://matplotlib.org/`.

[22] Kevin Arvai, Kyle Retterer, Vlad Gainullin, Carlos Borroto. phenopy 0.2.1, last visited: 1 March, 2020. `https://pypi.org/project/phenopy/0.2.1/`.

[23] Sebastian Köhler, Melissa Haendel, and Peter Robinson. Human phenotype ontology, last visited: 30 March, 2020. `https://hpo.jax.org/app/download/ontology`.

[24] M. Landrum, J. Lee, G. Riley, et al. *The NCBI Handbook*. Bethesda (MD): National Center for Biotechnology, 2013.

[25] Melissa J. Landrum, Jennifer M. Lee, Benson Mark, et al. ClinVar: Improving Access to Variant Interpretations and Supporting Evidence. *Nucleic Acid Research*, 46:D1062–D1067, 2018.

[26] Melissa J. Landrum, Jennifer M. Lee, Riley George R., et al. ClinVar: public archive of relationships among sequence variation and human phenotype. *Nucleic Acid Research*, 42:D980–D985, 2014.

[27] Stephen B. Montgomery, David L. Goode, Erika Kvikstad, et al. The origin, evolution, and functional impact of short insertion–deletion variants identified in 179 human genomes. *Genome Research*, 23:749–761, 2013.

[28] U.S. National Library of Medicine National Center for Biotechnology Information. Clinvar, last visited: 18 May, 2020. `https://www.ncbi.nlm.nih.gov/clinvar/`.

[29] NetworkX developers. Networkx, last visited: 20 May, 2020. `http://networkx.github.io/`.

[30] Beata Nowakowska. Clinical interpretation of copy number variants in the human genome. *Journal of applied genetics*, 58:449–457, 2017.

[31] U.S. National Library of Medicine. Oculocutaneous albinism, last visited: 1 May, 2020. `https://ghr.nlm.nih.gov/condition/oculocutaneous-albinism`.

[32] U.S. National Library of Medicine. How are gene mutations involved in evolution?, last visited: 27 April, 2020. `https://ghr.nlm.nih.gov/primer/mutationsanddisorders/evolution`.

[33] U.S. National Library of Medicine. What is a gene mutation and how do mutations occur?, last visited: 27 April, 2020. `https://ghr.nlm.nih.gov/primer/mutationsanddisorders/genemutation`.

[34] Stanley A. Sawyer, John Parsch, Zhi Zhang, and Daniel L. Hartl. Inaugural Article: Prevalence of positive selection among nearly neutral amino acid replacements in Drosophila. *Proceedings of the National Academy of Science*, 104(16):6504–6510, April 2007.

[35] David H. Spencer, Bin Zhang, and John Pfeifer. *Clinical Genomics*. Academic Press, 2015.

[36] Valentin Lab. Colour, last visited: 23 May, 2020. `https://github.com/vaab/colour`.

[37] M. Zarrei, J. MacDonald, D. Merico, et al. A copy number variation map of the human genome. *Nature Reviews Genetics*, 16:172–183, 2015.

[38] Chen Zhao, John Tynan, Mathias Ehrich, et al. Detection of Fetal Subchromosomal Abnormalities by Sequencing Circulating Cell-Free DNA from Maternal Plasma. *Clinical Chemistry*, 61:608–616, 2015.

Appendix: content of electronic appendix

In the electronic appendix of this work are included these file, which are also available on the `https://github.com/krisbal97/phency`.

- graphForGene.py — program implementing the gene model.
- graphForSequence.py — program implementing the CNV overlap model.
- complexGraph.py — program implementing the complex model.
- hpo.txt — file storing our adjusted version of HPO database in OBO format.
- cnvs_big.tsv — table storing information about CNV and their relation to phenotypes.
- genes_to_phenotype4.tsv — table storing information about gene–phenotype relationship.