MSc Research Paper

Predicting the thermodynamic effect of mutations on a genome-wide scale

Alexey Strokach

December 22, 2014

Abstract

Claim: $\forall n \in \mathbb{N}, P(n) \implies Q(n)$

1 Introduction

2 Methods

3 Results

4 Discussion

5 Future Directions

6 Energetic features

The computational methods for predicting protein stability upon mutation have been compared recently [**Potapov2009**].

Maximum correlation coefficient 0.86 [**Potapov2009**].

I-Mutant 2 and FoldX are trained

Rosetta can predict ddG as well, but the energy function has to be chosen carefully [**Kellogg2011**]. In particular, a soft repulsion energy function should be used for repacking (combinatorial rotamer optimization carried out using Monte Carlo simulated annealing with Dunbrack backbone dependent

rotamer library), optionally combined with a hard-repulsion energy function used during backbone and sidechain minimization with uniform constraints [**Kellogg2011**]. However, optimization leads to only slightly improved accuracy for a single mutation (0.68 vs 0.69 correlation coefficient), and can be skipped in order to speed up predictions.

When the reference energies for the 20 amino acids are fit to produce the best ddG correlation with experimental values, the correlation coefficient went up to 0.73. The source code for FoldX is not available. FoldX is trained on the Protherm and Skempi datasets, and therefore our cross-validation is likely overfitting the

7 Sequence features

Untrained:

SIFT Provean MutationAssessor FATHMM (unweighted)

Trained: Polyphen-2 MutationTaster FATHMM (weighted)

8 Co-evolution between amino acids

OMA

eggNOG

9 Co-evolution between amino acids

Using a balanced set of 6,630 mutations, the structure-based method results in about 3% higher accuracy and AUC and 0.06 higher correlation with respect to the sequence-based approach [**Capriotti2011**].

"Improving the prediction of disease-related variants using protein three-dimensional structure"

Figure 1: A schematic giving the overview of this project. **Left:** protein functional association network, formulated as a Merkov random field. Proteins connected by edges are predicted to have the same effect on the organism if either of the proteins is mutated. Proteins coloured a darker green are known to be involved in specific genetic diseases, and act as relays that convert genetic information into disease priors. **Centre:** the human disease ontology, formulated as a Bayesian network. The node at the top indicates the presence or absence of a given disease, and that node's parents provide an increasingly more specific diagnosis of the disease. **Right:** the human phenotype ontology, formulated as a Bayesian network. The node at the top corresponds to the presence or absence of a given phenotype (which could be any clinical finding), and that node's parents provide an increasingly specific description of that phenotype. Green dashed arrows link proteins in the protein network to diseases that those proteins may cause. Dashed blue arrows link diseases in the disease ontology to phenotypes that those diseases may cause. The shaded rectangle indicates the component of this gene-disease-phenotype network that we analysed in this project, and that we refer to as the gene-disease ontology (gene-DO) network.

Overview

The goal of this project was to implement a Bayesian network, which would link proteins in a protein association network with diseases in a disease ontology, and would permit the inference of diseases caused by the disruption of a given set of proteins, and inference of proteins whose disruption is likely to cause a given set of diseases. Such a Bayesian network could be used as a component of a larger clinical decision support (CDS) system, and could improve the accuracy of such a system by incorporating prior information based on the genomic sequence of the person being diagnosed. A general overview of the conceived CDS system, as well as the subcomponent of the system that is the focus of this project, is presented in Figure 1.

In brief, we construct a directed acyclic graph (DAG) that is based on the disease ontology, but also includes arcs from proteins to diseases, depicting known protein-disease associations. We convert this graph into a Bayesian network, by treating each non-terminal node as a noisy-OR gate, and by assigning to each disease a prevalence value that we obtain from electronic health record (EHR) and medical claims data. We compare the performance of different inference algorithms in this network, and describe a utility function that we could use to convert disease probabilities to diagnoses.

This report is divided into four sections. In the Introduction, we give a brief overview of CDS systems, focusing on the different methods that were used previously by other groups, and the corresponding benefits and shortcomings of those methods. We also describe in more depth the network that we construct in this project, and offer some justification and historic background for our choice of the network structure. In the Methods section, we describe how we constructed this network, listing our assumptions, our sources of data, and the problems that we encountered. In the Results section, we report the outcome of using different inference algorithms. In the Discussion section, we summarise what we have learned, list some topics that would be worth investigating in the future, and some ways in which we could make the network that we created better.

10 Introduction

10.1 Background on clinical decision support systems

Clinical decision support systems (CDS) are designed to aid the physician in making the correct diagnosis and prescribing the correct treatment for a given patient. CDS systems usually take as input the patient's history, physical examination results, laboratory test results, and other phenotypes, and provide as output a list of the most probable diagnoses [1].

Some of the first advanced CDS systems were INTERNIST-1/Quick Medial Reference (QMR) [2] (here referred to as QMR for brevity), and MYSIN [3], developed in the 1970s and 1980s by groups at the University of Pittsburgh and Stanford. Both systems were supported by domain expert knowledge, and used a rule-based strategy, modelled after the decision-making process of a physician, to make the diagnosis. The MYSIN system integrated expert knowledge into the diagnostic algorithms, which made the system difficult to maintain and update [1]. The QMR system, on the other hand, kept the knowledge-base decoupled from the decision algorithms, giving them more leeway to evolve independently.

Compiling the QMR knowledge-base (QMR-KB) took a large amount of effort, with over 20 man-years invested by the core domain export, and many contributions from students and collaborators. The goal was to create a complete "evidence-based, academic repository of diagnostic information", and by 1990 the QMR-KB consisted of over 600 diseases and 4000 findings, with on average of 85 findings and 8-10 linked diagnoses per disease [4].

The first diagnostic algorithms used by QMR linked every disease to every possible symptom, and this caused "program execution times [to grow] exponentially in the number of findings" [4] (a recurring topic in CSC2534). However, this rule-based algorithm had the advantage of listing the set of rules that led the algorithm to the particular diagnosis, which allowed the physician to accept or discredit the diagnosis based on his/her perceived credibility of the rules.

In the 1990s, Shwe *et.al.* reformulated the QMR diagnostic task as an inference problem in a probabilistic graphical network (QMR-DT) [5, 6]. Information extracted from the QMR-KB was used to construct a bipartite directed acyclic graph (DAG), which linked 534 diseases to 4,040 clinical findings using 40,740 arcs. The effect of multiple diseases on a single finding was modelled using noisy-OR gates, under the assumption that the effect of each disease is independent from the effects of all the other diseases. Equation 1 gives the probability of having a finding f, given a set of k disease parents $\{d_1...d_k\}$, that each cause the finding with a probability of $P(f = 1|d_i)$, computed using the noisy-OR assumption. The primary advantage of using noisy-OR gates is that they allow the construction of conditional probability tables (CPTs) from k rather than 2^k conditional probabilities.

$$P(f = 1|\{d_1...d_k\}) = 1 - \prod_{i=1}^{k}(1 - P(f = 1|d_i) \tag{1}$$

Shwe *et.al.* report that exact inference in the QMR-DT network was intractable when the network was queried with more than a handful of clinical findings. However, the authors were able to use heuristic

and simulation algorithms to perform approximate inference and to estimate QMR-DR prediction accuracy. Subsequent advances in variational inference techniques allowed for quicker and more accurate inference in the QMR-DT network [7–11].

There have been few advances in CDS systems since QMR-DT. Shwe *et.al.* mention that they plan to extend QMR-DT by adding utility and decision nodes, converting the Bayesian network into an influence diagram [5]. However, to our knowledge, that proposal had not been materialized. Promedas [12], one of the CDS systems that is currently on the market, uses the same probabilistic graphical model as QMR-DT, with the addition of a "risk factor" layer, which modifies the likelihood of different diseases based on risk factors. Inference in the Promedas network is performed using the junction tree algorithm and loop-corrected belief propagation, and there are cases for which it is reported to be intractable [12].

11 Methods

11.1 Building the gene-disease ontology network

The disease component of the gene-disease ontology (gene-DO) network was constructed by downloading and parsing the disease ontology (DO) .obo file, which produced a directed acyclic graph (DAG) with 6456 diseases and 6815 disease-disease arcs. We added to the graph 795 gene-disease associations that we extracted from DisGeNet [15] and HPO [16], and 2821 disease prevalence priors that we extracted from HuDiNe [17] and Stride [18], as described below. The overlap between all diseases in DO, diseases with gene-disease associations, and diseases with prevalence priors, is shown as a Venn diagram in Figure 2A.

After adding the gene-disease arcs and disease prevalence priors to DO, we iteratively removed all leaves in the ontology for which we had no prior information (*i.e.* we removed all diseases that had no gene-disease links and no disease priors for it or its parents), leaving us with a network that contained 3660 diseases (Figure 2B). Each disease that was not a leaf in the resulting network was given a conditional probability table (CPT) that assigned to it a probability of 0% if none of its parents were true, and a probability of 100% if any of its parents were true. This CPT is correct for diseases that only have other diseases as parents, as it correctly describes the structure of an ontology, but it is an oversimplification for diseases that also have proteins as parents, as it assumes that the disruption of those proteins will always cause the disease. We accept this oversimplification for this project, but note that in a production system we would have to construct more meaningful CPTs using actual mutation penetrance data.

In order to keep the sizes of the CPT tables reasonable, we limited the maximum number of parents that a node could have to 10. For nodes that had more than 10 parents, we introduced dummy nodes that would each integrate the signal from a subset of the parents, using the noisy-OR formulation and the same probabilities as the original node. The original node would then combine the signal from a much smaller number of dummy nodes.

Figure 2: Venn diagrams showing the fraction of diseases in the disease ontology (blue) that were covered by gene-disease association (cyan) and gene prevalence (magenta) data. A shows all diseases in the disease ontology. B shows the diseases in the disease ontology that remained after the ontology was pruned to remove leaves (diseases) with no prior information.

For diseases that were the leaf nodes of the constructed network, we used probabilities that we obtained from the HuDiNe and Stride databasets. For proteins, we used a simple "placeholder" probability of 0.00053, which we obtained by dividing a probability of 1.0 by the number of proteins in the network. Ultimately, we would like to obtain a personalised probability that the function of a given protein is disrupted by analysing each individual's genome sequencing data and extrapolating the results to proteins that are known to be involved in disease (left panel of Figure 1).

A considerable simplification that we made while creating the network was to assume that the probability of each non-terminal node is determined by its parents (*i.e.* that a disease is only be present if any of its parent disease are present or any of its parent proteins are mutated). In many cases, a patient may be diagnosed with a disease that is not a leaf in the disease ontology, and the prevalence of those diseases would be underestimated by the current assumption. To address this issue, we could either distribute the "surplus probability" of the disease equally among its parents, or create a new "leaky node" parent that gives the disease a certain probability of being true even when all of its parents are false. We will likely use the latter approach in our future work.

The network was constructed, as described above, using the python programming language. It was exported using the Hugin .net file format, which is supported by many software packages designed for working with PGMs: Hugin, Genie, SamIam, UnBBayes, gRain, and others.

11.1.1 Parsing gene-disease association data

DisGeNet [15] is a comprehensive database of gene-disease associations. It integrates gene-disease associations curated by experts, with gene-disease associations inferred by homology and by text-mining the scientific literature. All gene-disease associations are assigned a confidence score, which describes the strength of the supporting evidence, and one of three association types, which describes the role that the given gene plays in the disorder. We considered only the *GeneticVariation* association type (DisGeNet-gv), as this was deemed to be the only type relevant when using genetic variants to predict disease.

The human phenotype ontology (HPO) "aims to provide a standardized vocabulary of phenotypic abnormalities encountered in human disease" [16]. The HPO developers made available an extensive list of gene-disease-phenotype associations, which they created by processing and extending the information in OMIM [19] and Orphanet [20]. We used the gene-disease subset of this file, here referred to as HPO-D, as another source of curated, high-confidence gene-disease associations.

While DO uses it's own disease identifier system, it provides cross-references to the identifiers used by OMIM, Orphanet, international classification of disease 9 (ICD-9), and NIH Unified Medical Language

Table 1: The fraction of diseases present in the DisGeNet-gv and HPO-D gene-disease association datasets that could be mapped to the disease ontology.

	DisGeNet-gv	HPO-D	Total	Overlap
Number of unique genes	1,839	3,158		
Number of unique diseases	2,376	4,919		
Number of unique gene-disease pairs	2,622	6,481		
Number (and %) of diseases that were mapped to the disease ontology	233 (10%)	782 (16%)	795	220

System (CUI). We used those cross-references to map to DO the CUI identifiers adapted by Disgenet and the OMIM and Orphanet identifiers adapted by HPO. Table 1 shows the surprisingly low success rate the we achieved when mapping those disease identifiers. The HPO-D dataset was actually introduced at a later stage of this project, in order augment the small number of gene-disease associations that we extracted from DisGeNet, and to see if the low mapping rate is specific to the CUI identifiers used by Disgenet. While including HPO did introduced a substantial number of new gene-disease associations (Table 1), the mapping success rate for HPO was not distinctly higher (10% vs. 16%).

11.1.2 Parsing disease prevalence data

The human disease network (HuDiNe) [17] contains disease occurrence and co-occurrence data extracted from Medicare inpatient claims of 32 million Americans aged 65 or older over a 3-year period. The Stanford center for clinical informatics (Stride) [18] dataset contains clinical concept occurrence and co-occurrence frequencies extracted from electronic health records (EHR) of 1.2 million Americans over a 19-year period. We mapped the ICD-9 disease ids used by HuDiNe, and the CUI ids used by Stride, to the unique disease ids used by DO using DO cross-references, and we combined the two disease prevalence datasets to obtain disease prevalence priors for 2821 diseases (Table 2). The fraction of disease identifiers that could be mapped to the DO was low, and was similar to the fraction of disease identifiers that could be mapped in the gene-disease datasets (Table 1).

In a production environment, we would group disease prevalence data by sex, age, ethnicity and possibly other variables describing each individual, and we would create different disease priors for different population demographics. However, in order to keep things simple for this project and maximize the coverage of diseases in the DO, we simply combined the disease prevalence data from the two sources, taking the average disease prevalence when the same disease was annotated by both the HuDiNe and the Stride dataset.

12 Results

12.1 Performing inference

Our first step in analysing the gene-DO network was to see if we can perform inference successfully using one of the three commonly-used software packages for PGMs: the gRain [21] and bnlearn [22] plugins for R, SamIam [23], and UnBBayes [24].

Table 2: The fraction of diseases present in the HuDiNe and Stride disease prevalence datasets that could be mapped to the disease ontology.

	Stride	HuDiNe	Total	Overlap
Number of patients	1.2 million	13 million		
Number of unique diseases ('concepts')	18,787	15,983		
Number (and %) of diseases that were mapped to the disease ontology	1635 (8.7%)	1888 (11.8%)	2821	702

The most popular exact inference algorithm is the junction tree algorithm, which is a variant of the variable elimination algorithm that uses junction trees in order to speed up the execution of repeated queries [25]. The junction tree algorithm is implemented in all three packages, and in all three packages attempting to use it to perform inference in the gene-DO network failed some form of the "out of memory" error, even when using a cluster with 250 GB of RAM. The gRain package gave the most informative error:

Thus, it appears that the junction tree algorithm requires too much memory to work with the gene-DO network on most current systems.

Next we tried the recursive conditioning algorithm, implemented in SamIam. Recursive conditioning is an "anyspace" algorithm that can perform exact inference on a Bayesian network using any amount of memory, taking longer to perform inference the less memory is specified [25]. Despite the claim to be "anyspace", the recursive conditioning algorithm in SamIam failed to work with our network, giving "overflow" errors even when it was instructed to use the least amount of memory possible.

Given our lack of success with exact inference algorithms, we tried to use some of the approximate learning algorithms, hoping to achieve more promising results. The bnlearn package can calculate the conditional probability of an event given evidence, and the conditional probability tables of a set of nodes given evidence, using either logic sampling or likelihood weighting sampling. However, it repeats the sampling for every query, which makes it inefficient and time-consuming when performing a large number of queries.

UnBBayes implements Gibbs sampling in addition to logic sampling and likelihood weighting sampling, and allows the user to save the output of the sampling algorithms to a text file. Logic sampling and likelihood sampling algorithms produced over 100,000 samples in several hours, but given the low probability of many of the proteins and diseases in our network, many more samples would likely be required to accurately estimate conditional probabilities. The Gibbs sampling algorithm produced less than 10,000 samples in the same amount of time.

SamIam was the only package that implemented loopy belief propagation and edge deletion belief propagation algorithms. The loopy belief propagation algorithm was able to calculate the marginal probability of each node in the network in several seconds, and was able to calculate conditional probabilities

given evidence in under a few minutes for many of the queries that we tried. However, in some cases, it failed to converge. Edge deletion belief propagation took substantially more time; calculating the marginal probabilities in a network without any evidence took several minutes with an edge recovery rate of 40 out of 1011.

Overall, loopy belief propagation is the most tractable algorithm for the gene-DO network, out of the algorithms that we tried. Variational inference methods [8, 9] possibly could have led to better performance, but we could not find a package that could run variational inference on our network (and did not have time to implement one on our own).

12.1.1 Learning network structure and parameters

We explored the possibility of refining the structure and parameters of our network using electronic medical record data. The ideas of this was appealing because the complex patterns of disease co-occurrence likely carry information about unknown protein-disease associations (assuming that at least some disease co-occurrence is due to a common genetic cause). However, we discovered several difficulties that prevented us from following through with this goal:

1. Due to privacy reasons, EMR data is easily accessible only in it's parsed form, as disease occurrence and disease co-occurrence frequencies. The frequency of co-occurrence for more than two diseases is lost, and we would have to reconstruct this information using some assumption. The simplest assumption would be say that the frequency with which two disease co-occur is never influenced by the existence of a third disease.

2. While several methods have been described for performing parameter learning in Noisy-OR networks [26, 27], those methods have not been implemented in an accessible package, and implementing a method ourselves would be time-consuming without necessarily being fruitful.

3. If we don't constrain the CPT tables using the noisy-OR assumption, we would be faced with a large number of parameters that would be difficult to learn and to interpret. Furthermore, we were not able to find a package that could refine a subset of arcs and CPTs in a Bayesian network using data with missing values.

12.2 The utility of probabilities

Due to time constraints, defining a utility function that would take as input a set of conditional probabilities, and would output a ranked list of diagnoses, was delegated to a later project. However, we can describe some qualities that such a utility function would possess:

1. If a disease has the same probability of occurring as its parent, we would always rank the parent higher in the final list of diagnoses. (In practice, the probability of the parent would be increased by some tunable parameter ϵ, in order to account for the fact that the probability of a child will always be higher than the probability of it's parent for any child that has more than one parent, as none of the nodes in our network have 0 probability).

2. If a node has several parents that have the same probability, we would rank that node ahead of its parents. This is because selecting one of the parents would carry a high risk of selecting the wrong parent, and we are assuming that our agent is risk averse.

13 Discussion

Depend on closed-source FoldX. Can use openMM to recreate most of the features that are used by FoldX and train a final classifier using those features.

Can use thermodynamic integration (TI) to increase the training set. Select a few mutations deemed to be the most important from each domain family.

References

1. Moore, M; Loper, KA An Introduction to Clinical Decision Support Systems. *Journal of Electronic Resources in Medical Libraries* **2011**, *8*, 348–366.
2. Myers, JD In *Proceedings of ACM Conference on History of Medical Informatics*, ACM: New York, NY, USA, 1987, pp 195–197.
3. Buchanan, BG; Shortliffe, EH, Rule Based Expert Systems: The Mycin Experiments of the Stanford Heuristic Programming Project (The Addison-Wesley Series in Artificial Intelligence); Addison-Wesley Longman Publishing Co., Inc.: Boston, MA, USA, **1984**.
4. Miller, RA A history of the INTERNIST-1 and Quick Medical Reference (QMR) computer-assisted diagnosis projects, with lessons learned. *Yearbook of Medical Informatics* **2010**, 121–136.
5. Shwe, MA; Middleton, B; Heckerman, DE; Henrion, M; Horvitz, EJ; Lehmann, HP; Cooper, GF Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. I. The probabilistic model and inference algorithms. *Methods of Information in Medicine* **1991**, *30*, 241–255.
6. Middleton, B; Shwe, MA; Heckerman, DE; Henrion, M; Horvitz, EJ; Lehmann, HP; Cooper, GF Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. II. Evaluation of diagnostic performance. *Methods of Information in Medicine* **1991**, *30*, 256–267.
7. Heckerman, D In *In UAI-89*; 00000, 1989, pp 163–172.
8. Jaakkola, TS; Jordan, MI Variational Probabilistic Inference and the QMR-DT Network. *Journal of Artificial Intelligence Research* **1999**, *10*, 291–322.
9. Ng, AY; Jordan, MI Approximate inference algorithms for two-layer Bayesian networks. *NIPS* **1999**, 200–.
10. Kappen, HJ; Wiegerinck, W In *Advances in Neural Information Processing Systems 14*; 00000, MIT Press: 2001, pp 415–422.

11. Heskes, T; Zoeter, O In *In Proceedings UAI*; 00092, 2002, pp 216–223.

12. Wemmenhove, B; Mooij, JM; Wiegerinck, W; Leisink, M; Kappen, HJ; Neijt, JP In *Artificial Intelligence in Medicine*, Bellazzi, R, Abu-Hanna, A, Hunter, J, Eds.; Lecture Notes in Computer Science 4594, 00022; Springer Berlin Heidelberg: 2007, pp 456–460.

13. Zemojtel, T et al. Effective diagnosis of genetic disease by computational phenotype analysis of the disease-associated genome. *Science Translational Medicine* **2014**, *6*, 252ra123–252ra123.

14. Vanunu, O; Magger, O; Ruppin, E; Shlomi, T; Sharan, R Associating Genes and Protein Complexes with Disease via Network Propagation. *PLoS Comput Biol* **2010**, *6*, e1000641.

15. Bauer-Mehren, A; Rautschka, M; Sanz, F; Furlong, LI DisGeNET: a Cytoscape plugin to visualize, integrate, search and analyze gene–disease networks. *Bioinformatics* **2010**, *26*, 2924–2926.

16. Kohler, S et al. The Human Phenotype Ontology project: linking molecular biology and disease through phenotype data. *Nucleic Acids Research* **2013**, *42*, D966–D974.

17. Hidalgo, CA; Blumm, N; Barabási, AL; Christakis, NA A Dynamic Network Approach for the Study of Human Phenotypes. *PLoS Comput Biol* **2009**, *5*, e1000353.

18. Finlayson, SG; LePendu, P; Shah, NH Building the graph of medicine from millions of clinical narratives. *Scientific Data* **2014**, *1*, 140032.

19. Hamosh, A; Scott, AF; Amberger, JS; Bocchini, CA; McKusick, VA Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. *Nucleic Acids Research* **2005**, *33*, D514–D517.

20. Rath, A; Olry, A; Dhombres, F; Brandt, MM; Urbero, B; Ayme, S Representation of rare diseases in health information systems: The orphanet approach to serve a wide range of end users. *Human Mutation* **2012**, *33*, 803–808.

21. Højsgaard, S Graphical Independence Networks with the gRain Package for R. *Journal of Statistical Software* **2012**, *46*, 1–26.

22. Scutari, M Learning Bayesian Networks with the bnlearn R Package. *Journal of Statistical Software* **2010**, *35*, 1–22.

23. Gaag, LCvd; Renooij, S; Coupé, VMH In *Advances in Probabilistic Graphical Models*, Dr, PL, Dr, JAG, Dr, AS, Eds.; Studies in Fuzziness and Soft Computing 214, 00024; Springer Berlin Heidelberg: 2007, pp 103–124.

24. Matsumoto, S; Carvalho, RN; Ladeira, M; Costa, P; Santos, L; Silva, D; Onishi, M; Machado, E, UnBBayes: a Java Framework for Probabilistic Models in AI,

25. Darwiche, A, Modeling and Reasoning with Bayesian Networks; Cambridge University Press: Cambridge, **2009**.

26. Halpern, Y; Sontag, D, Unsupervised Learning of Noisy-Or Bayesian Networks,

27. Jernite, Y; Halpern, Y; Sontag, D In *Advances in Neural Information Processing Systems 26*, Burges, CJC, Bottou, L, Welling, M, Ghahramani, Z, Weinberger, KQ, Eds., 00002; Curran Associates, Inc.: 2013, pp 2355–2363.

28. Pedregosa, F et al. Scikit-learn: Machine Learning in Python. **2012**.

29. Chen, YC; Douville, C; Wang, C; Niknafs, N; Yeo, G; Beleva-Guthrie, V; Carter, H; Stenson, PD; Cooper, DN; Li, B; Mooney, S; Karchin, R A Probabilistic Model to Predict Clinical Phenotypic Traits from Genome Sequencing. *PLoS Comput Biol* **2014**, *10*, e1003825.

30. Church, GM The Personal Genome Project. *Molecular Systems Biology* **2005**, *1*, 2005.0030.