Modelling Autism Genomics in Families Using Heterogeneous Graph Neural Networks

Sameer Sardaar
Department of Human Genetics
McGill University
Montreal, Canada
sameer.sardaar@mail.mcgill.ca

Recep Yusuf Bekci
Desautels Faculty of Management
McGill University
Montreal, Canada
recep.bekci@mail.mcgill.ca

Abstract

We model genomic mutations in families with at least one child with Autism Spectrum Disorder (ASD) in a graph, and propose a heterogeneous graph neural network (GNN) variant to perform a node classification task. We provide an extensive literature review of the work done on heterogeneous GNNs and extend their application to our novel problem of genetic disorder prediction in families with one affected child with ASD. This study aims at predicting ASD status (i.e. affected or unaffected) in children based on their and their parents' rare protein-coding mutations.

1 Background and motivation

Autism spectrum disorder (ASD) is a heterogeneous group of neurodevelopmental disorders affecting 1 out of 68 children, with heritability estimated from twin studies to be around 90% Bourgeron [2016] meaning most of the variance in the general population can be explained by genetic factors. Children with ASD show deficits in communication and social behaviour and have repetitive behaviour Bourgeron [2016]. Genome-wide studies have revealed mutations in large number of susceptibility genes (100-1000) to be associated with ASD Griswold et al. [2015] Iossifov et al. [2014] Iossifov et al. [2012]. However, each individual mutation in the identified genes has a very small effect size and when taken together, can only explain a very small amount of phenotypic risk.

The range of cumulative risk explained by the weighted sum of all the identified mutations using Polygenic Risk Scoring (PRS) method fall between 0.5% to 0.7% Jansen et al. [2019]. This is extremely low given the high estimated heritability of Autism. This phenomenon in the field of genetics is known as "missing heritability". It is the unexplained variance (i.e. gap) between heritability estimated from twin studies and its proportion estimated from genotype data through genome wide association studies (GWAS).

The lack of success is thought to be due to ASD having polygenic architecture, where each susceptibility gene is increasing one's predisposition to the disease in combination with other genes (i.e. gene to gene interactions). However, the exact interaction mechanism is unknown. Early diagnosis of ASD based on an accurate predictive algorithm would have implications for preventing the disease from occurring in a family (using preimplantation or prenatal genetic options) and/or would provide the potential for early treatment initiation that could decrease the severity of ASD in affected individuals.

In this project, we work with whole exome sequenced data of affected and unaffected family members and represent the data in a heterogeneous network where we have two types of nodes (humans and SNVs) and two types of edges ("human - SNV link" and "child-parent relationship"). Our prediction task is the disease status of "human" children nodes. Single nucleotide polymorphisms (SNVs) are the most common type of mutations in the genome where a single nucleotide in a stretch of DNA is

mutated. Throughout this report, we use the terms SNV, variant, and mutation interchangeably in the context of genetics.

To the best of our knowledge, there has been no prior work done on modelling ASD in families based on their genomic profiles in a graph. However, in one related work done by Parisot et al. [2018], they conduct a node classification task (ASD status) on a population graph using graph convolutional neural networks leveraging brain images and phenotypical data as node attributes. Unlike our graph, they connect nodes in their graph according to some similarity metrics they defined. Another related work by Schulte-Sasse et al. [2019] deals with genomic data in a graph but uses graph convolutional neural networks to predict cancer driver genes as opposed to disease status prediction.

2 Proposed approach

In this paper, we investigate the use of graph neural network approaches on homogeneous and heterogeneous graphs for downstream node classification task. Traditional deep learning methods have been quite successful at learning latent representation of euclidean data but cannot generalize to graphs which are irregular objects, and important notions such as convolution are not well-defined on them Wu et al. [2019]. However, there has been a significant amount of research done lately to extend neural networks to graph structured data and generalize the convolution operation to graphs. One main concept behind these methods is the recursive propagation and aggregation of feature information from node neighborhoods using neural networks Schlichtkrull et al. [2018], Veličković et al. [2017], Hamilton et al. [2017]. However, most of these methods are focused on homogeneous graphs where there is only one type of edges and nodes.

In an effort to generalize GCNs to relational or heterogeneous graphs, Schlichtkrull et al. [2018] proposed Relational GCNs (R-GCNs) which extends the GCN convolutional operator and the authors proposed to apply relation type specific transformations to the message-passing framework of GCNs. The authors demonstrate the effectiveness of their model on both entity classification and link prediction tasks. Similarly, in order to model polypharmacy side effects, Zitnik et al. [2018] proposed Decagon which implements an R-GCN variant on heterogeneous drug-drug and protein interaction networks to perform link prediction task. The convolution operator used in Decagon encoder aggregates feature information across different types of edges. The authors further demonstrate that Decagon outperforms all other existing approaches to the problem.

In another work by Zhang et al. [2019], they propose HetGNN (Heterogeneous Graph Neural Network) which samples different node type neighbors using random walk with restart and collects fixed length sequences. Then, HetGNN performs encoding and aggregation per node type using neural networks. Another work by Wang et al. [2019a] leverages the concept of metapaths and proposes a heterogeneous graph attention network with node-level and semantic-level attentions. They employ deep neural networks to learn node level attentions, and weights for semantic-specific metapaths to get final embedding of a node. Similarly, Ren et al. [2019] proposes HDGI (Heterogenous Deep Graph Infomax), which uses meta-path structure to analyze connections and use them for learning local representations, and then aggregates these representations using a semantic level attention mechanism to get a global representation.

2.1 Methodology

As a general methodology, we are planning to follow a semi-supervised approach to train GNNs on our graph to conduct node classification task. For some implementations, we will leverage existing Deep Graph Library Wang et al. [2019b] and PyTorch Geometric Fey and Lenssen [2019] libraries. As our baselines, we will train a graph convolutional neural network on homogeneous variant of our data and gradient boosting machine model on the concatenated set of features only. This concatenation process makes parental information available for each sample (i.e. child) in a tabular format. As a result, in families with one affected child and one unaffected, we end up having two samples with the same parental features for the XGBoost model.

In our proposed GNN model, we extend the GCN baseline to take our heterogeneous graph as input, and perform semi-supervised node classification task in an end-to-end manner. We split our target nodes (i.e. 4192 children) in the graph into training, validation and testing sets of 70:15:15 ratio, where we mask the labels of validation and testing sets during training.

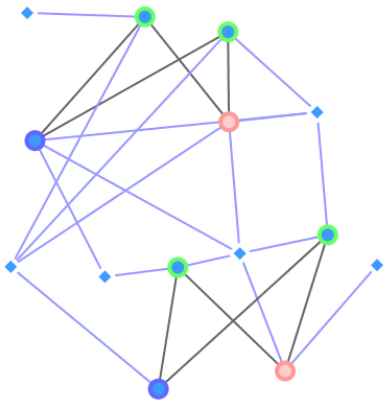

Figure 1: A mini example representation of our graph with two families and a few SNVs. Diamond-shaped nodes denote SNV mutations and ellipse-shaped nodes denote human nodes. Parents are denoted with green, affected child with red and unaffected child with blue border paint. Human to SNV link is denoted with blue edge type, and child-parent relationship with black-coloured edge type.

3 Data availability and computational requirements

The data for 2,392 families (8,976 individuals) with at least one affected child with ASD were obtained from the National Database of Autism Research. The raw exome sequencing data is of families in the Simons Simplex Collection. One of the requirements for participation in the study was that both biological parents had to be willing to participate and that they should not have ASD. Of the 2,392 families, 1,800 of the families are quads (i.e. unaffected parents with one affected and one unaffected child) and 592 of them trios (i.e. unaffected parents with one affected child). We denote affected and unaffected children with a binary label.

From the original sequencing data, we filter for rare and functional variants in the exome, resulting in a total of around 140,000 SNVs. Therefore, our entire network ends up having close to 149,000 nodes and around 5,9000,000 edges. Each SNV contains features such as variant type, functional type, genotype, and minor allele frequency. Categorical features of SNVs are turned into one-hot encoding which results in around 1,000 long feature vector per SNV node. Each human node has "parent", "child", "affected" and "unaffected" status features. For computation, we need to utilize GPUs. Considering the complexity of our graph and types of models under study, computation resources available to us through Compute Canada should suffice.

3.1 Data preparation and pre-processing

As a major initial step for our project, we created a data pipeline to clean, filter and select the variants from the whole exome sequencing files of each chromosome. Then we turned the cleaned data to a tabular, homogeneous and heterogeneous graph objects for our baselines and proposed model.

From the exome files, we selected for coding variant types {frameshift deletion, frameshift insertion, frameshift substitution, nonsynonymous SNV, stop gain, or stop loss} and variant functional types annotated as one of {exonic, exonic splicing, intronic splicing}. Furthermore, we selected for rare variants, meaning variants which have minor allele frequency (MAF) equal to or less than 0.01. And on a per-individual basis, for variants to be called they needed to have a minimum number of 4 variant reads, minimum depth of sequencing of 10 reads, and a minimum genotype quality of 90. We also excluded trizygotes from our data, and named each variant by a proxy ID as their identifier following the Chromosome-Position-Reference allele-Mutated allele nomenclature.

4 Experimental setup

We set up our problem as a node classification task where we are interested in predicting the disease status of human children nodes (i.e. binary classification) in our proposed graph. To this end, we divide our target nodes (i.e. 4192 children) in the graph into training, validation and testing sets of 70:15:15 ratio, where we mask the labels of validation and testing sets during training. The validation set at this stage is not being utilized for optimizing any hyperparameters and is mainly intended for later work. We optimize our model using the cross-entropy loss where in the binary classification task takes the form of:

$$J(w) = -(ylog(p) + (1-y)(log(1-p))$$

We evaluate the performance of our models using the accuracy metric. Given that our dataset is only slightly unbalanced between positive and negative samples, accuracy is still an appropriate measure to use and we are able to prevent unbalanced prediction behaviour with basic modifications in our loss function.

4.1 Baseline models

We use gradient boosting machine algorithm as a baseline model and utilize its regularized and efficient XGBoost implementation. We train the model on concatenated set of children and their parents' features. We used 500 trees and stopped the training process after validation loss started to decrease after three iterations. We didn't conduct a complete fine tuning process on the model.

As a second baseline model, we train a graph convolutional neural network (GCN) variant on our graph. Since a standard GCN model is not designed for handling more than one node and relation types, we model our data as a homogeneous graph, where there is only one type of nodes and relations. In this type of graph, we assign node type as an attribute of the nodes. In a single GCN message passing layer, node features are aggregated and transformed by some linear transformation which then goes through a non-linearity. The updated representation in the next layer can be summarized in the following equation:

$$h_v^{t+1} = \sigma\left(\sum_{u \in N_v} W^{(t)} h_u^{(t)}\right)$$

where $W^{(t)}$ is the weight matrix at layer t. $h^{(t)}$ is the output of the layer t and N_v is the neighbourhood nodes of node v. $\sigma(\cdot)$ applies some non-linearity on the output such as a rectified linear unit. We add self-connection to all nodes to make sure their features are aggregated with the receiving messages in a single update layer.

Our Convolutional Graph Neural Network consists of two layers similar to the architecture proposed in the original [Kipf and Welling, 2016] paper. In our graph, we also need to propagate a maximum of two layers so a child node can capture the genomic profile of its parents. In our GCN experiments, it was challenging to improve the accuracy over majority class prediction so we supplied class weights into the loss function. We calculated the class weights using the heuristic described in King and Zeng [2001]. After introducing class weights to the model, it started predicting fairly between the two classes, but started to overfit quickly and we would not observe any improvement over the random prediction case.

We addressed over-fitting by adding l_2 regularization and dropout layers. When we apply them separately, we observed a delay in the over-fitting process without any added improvement on validation accuracy. In addition, we experimented with batch-normalization layers after the hidden layer. This experiment resulted in an instability of training process and we observed negative loss. We used 0.2 dropout rate and $relu$ activation functions. For the learning rate, we tried values $[0.0001, 0.001, 0.01]$. We observed instability with 0.01 similar to the case of batch-normalization. For the hidden layer size, we observed exactly the same performance for 16, 64. However, the model with 132 hidden nodes would tend to over-fit very quickly. For the final baseline model, we used 0.0001 learning rate with 64 hidden nodes.

4.2 Proposed model

We extend the GCN model from above and turn our data into a heterogeneous graph where different relation and node types are preserved. Therefore, motivated by the R-GCN model, we implement relation-specific transformations using separate MLPs per canonical relation type. Therefore, each node's feature content and embeddings get transformed and reduced differently based on the type and direction of the relations on which it's being propagated.

As a result, our model's propagation layer takes the following form:

$$h_v^{t+1} = \sigma\left(\sum_{r \in R} \sum_{u \in N_v^r} MLP_r^{(t)}(h_u^{(t)})\right)$$

Where N_v is denoting the neighborhood of some node v and R = {SNV-exists in-Human, Human-has-SNV, SNV-self-SNV, Human-parents-Human, Human-self-Human} and MLP a multi-layered neural network with dropout. Each relation-specific MLP consists of two layers with a dropout and ReLU activation function after the first layer. Unlike the original R-GCN model proposed in Schlichtkrull et al. [2018] paper, we use MLPs instead of some linear transformation and train our network on both node embeddings and features.

We use the mean function for relation specific reduction and sum for aggregation at the node level. For learning, we stack two layers of our proposed model and optimize over all parameters in an end-to-end manner while using cross-entropy loss function with Adam optimizer. Given that we have a slight class imbalance in the data, we weigh the minority class accordingly similar to the GCN approach. We train our final model with a learning rate of 0.001 and set weight decay for regularization to 0.0001. We also set dropout rate to 0.2 in the dropout layer of the relation specific MLP encoders. We stop training if validation accuracy does not improve after 3 epochs. However, when we left the model to train for longer, it started to overfit after 100 epochs, and validation accuracy started decreasing.

In addition to node features, we also add node IDs as learnable embeddings in our GNNs so they can learn important topological structures in the graph. For example, learning the difference between de-novo and inherited mutations in the graph is very important which form different shapes in the graph with human nodes. A likely inherited mutation from a parent will form a triangle with the child and parent nodes, or two of them in the case of most likely inherited SNV where both parents and child have the same mutation. This concept is illustrated in figure 2.

Lastly, given that about 25% of our families are trio sets (i.e. unaffected parents + affected child) and the other 75% quads (i.e. unaffected parents + affected child + unaffected child), we did not want our GNN to learn the family unit structure as predictive of diseases status. For example, it could falsely learn that trio family structure in the graph is predictive of disease status, and would falsely get the prediction right. Therefore, we made the child-parent relationship uni-directional going from parent node to child node. As a result, there is no flow of sibling information through parent relationship while we propagate messages within 2 hop neighborhood of a child node.

5 Preliminary results

We trained the XGBoost baseline model on family-wide concatenated features in a tabular format, and our GNN models on homogeneous or heterogeneous graphs with node embeddings and features. The performance of the three models is summarized in Table 1.

The GNN models outperformed XGBoost which is using concatenated features only. It has higher validation accuracy compared to the GNNs so it's possible that XGBoost is suffering from slight overfitting. It's worth noting that by predicting the majority class alone, a model could achieve a test accuracy of 58.12% but we have weighed the classes accordingly to prevent this from happening. Nonetheless, this is close to where both GNN models end up converging with our proposed R-GCN performing slightly better at 58.44% compared to GCN at 58.28%, outperforming random predictions.

Model	Training Accuracy	Validation Accuracy	Test Accuracy
XGBoost	0.6685	0.5707	0.5621
GCN	0.5702	0.5682	0.5828
R-GCN	0.5721	0.5517	0.5844

Table 1: A summary of model performance: The proposed R-GCN variant with MLP relation specific encoders outperform both baselines achieving 58.44% test accuracy.

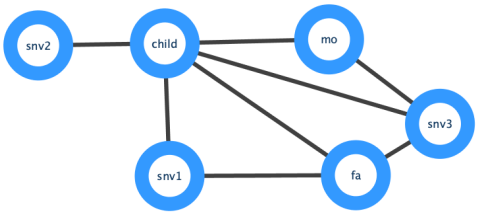

Figure 2: A small homogeneous example of our large heterogeneous graph shown to demonstrate some important topological structures in the graph that translate to important genetic concepts. It's crucial in genetics to understand the possible source (i.e. inheritance pattern) of each mutations in one's genome and how it compares between affected and unaffected children across a population. The graph here is showing the difference among de novo mutation, possibly inherited and most likely inherited mutations by SNV2, SNV1 and SNV3 respectively. They can be distinguished by the triangle(s) pattern formation, or there lack of, between the child and each parent with a given SNV.

6 Discussion and next steps

We modelled the prediction of ASD in children as a node classification task using graph neural networks on rare genomic variants of families. The test accuracy of our proposed model is low but not surprising. It confirms the existing Polygenic Risk Scoring (PRS) results for ASD which at the moment explain 0.5% to 0.7% of phenotypic variance in the population. PRS predictions are based on the weighted sum of univariate statistical association tests done through genome wide association studies on unrelated cases and controls. It does not take into account the epistatic interaction of variants, their heritability pattern or any other features of a variant. By modelling our data in a graph, we addressed these limitations and put every child's genomic profile within the context of their genetic code source (i.e. parents) and variant features. As most of our samples are siblings who share most of their genome variants, it makes our prediction task harder but our results more genetically robust as they are not impacted by any genetic bias or population stratification.

Nevertheless, there are some next steps for us to explore in the future. For example, we need to enrich our graph with existing knowledge regarding variant pathogenicity, and also incorporate protein-protein interaction results using STRING database API Szklarczyk et al. [2019] to further connect our graph and to add the biological dimension of gene interaction. This will enable our GNN to put mutations within the context of their gene (i.e. function) and gene product (i.e. proteins). Also, since ASD is a brain disorder, weighing SNVs in genes with high functional impact on the brain should be explored as well.

In addition, there are other areas of improvements in our proposed GNN architecture to explore in the future. For example, we were planning to add an attention mechanism to aggregate type-specific reduced messages at the nodes. These type specific attention weights can be trainable weight vectors or be motivated by genetics. Lastly, a mutation can occur or get inherited in different zygosity in humans, meaning whether it occurs in one allele (heterozygous: one hit) or both alleles (homozygous: two hits) in a chromosomal location. Therefore, it should be modelled as an edge attribute and its proper representation learned with its corresponding SNV in a Heterogeneous or Relational GNN. These improvements will require further extension of our proposed GNN model to learn both edge representations and attention weights to model the complex genetics of ASD.

References

Thomas Bourgeron. Current knowledge on the genetics of autism and propositions for future research. *Comptes rendus biologies*, 339(7-8):300–307, 2016.

Anthony J Griswold, Nicole D Dueker, Derek Van Booven, Joseph A Rantus, James M Jaworski, Susan H Slifer, Michael A Schmidt, William Hulme, Ioanna Konidari, Patrice L Whitehead, et al. Targeted massively parallel sequencing of autism spectrum disorder-associated genes in a case control cohort reveals rare loss-of-function risk variants. *Molecular autism*, 6(1):43, 2015.

Ivan Iossifov, Brian J O'roak, Stephan J Sanders, Michael Ronemus, Niklas Krumm, Dan Levy, Holly A Stessman, Kali T Witherspoon, Laura Vives, Karynne E Patterson, et al. The contribution of de novo coding mutations to autism spectrum disorder. *Nature*, 515(7526):216–221, 2014.

Ivan Iossifov, Michael Ronemus, Dan Levy, Zihua Wang, Inessa Hakker, Julie Rosenbaum, Boris Yamrom, Yoon-ha Lee, Giuseppe Narzisi, Anthony Leotta, et al. De novo gene disruptions in children on the autistic spectrum. *Neuron*, 74(2):285–299, 2012.

Arija G Jansen, Gwen C Dieleman, Philip R Jansen, Frank C Verhulst, Danielle Posthuma, and Tinca JC Polderman. Psychiatric polygenic risk scores as predictor for attention deficit/hyperactivity disorder and autism spectrum disorder in a clinical child and adolescent sample. *Behavior genetics*, pages 1–10, 2019.

Sarah Parisot, Sofia Ira Ktena, Enzo Ferrante, Matthew Lee, Ricardo Guerrero, Ben Glocker, and Daniel Rueckert. Disease prediction using graph convolutional networks: application to autism spectrum disorder and alzheimer's disease. *Medical image analysis*, 48:117–130, 2018.

Roman Schulte-Sasse, Stefan Budach, Denes Hnisz, and Annalisa Marsico. Graph convolutional networks improve the prediction of cancer driver genes. In *International Conference on Artificial Neural Networks*, pages 658–668. Springer, 2019.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer, 2018.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017.

Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466, 2018.

Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 793–803, 2019.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The World Wide Web Conference*, pages 2022–2032, 2019a.

Yuxiang Ren, Bo Liu, Chao Huang, Peng Dai, Liefeng Bo, and Jiawei Zhang. Heterogeneous deep graph infomax. *arXiv preprint arXiv:1911.08538*, 2019.

Minjie Wang, Lingfan Yu, Da Zheng, Quan Gan, Yu Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Qi Huang, Chao Ma, et al. Deep graph library: Towards efficient and scalable deep learning on graphs. *arXiv preprint arXiv:1909.01315*, 2019b.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Gary King and Langche Zeng. Logistic regression in rare events data. *Political analysis*, 9(2): 137–163, 2001.

Damian Szklarczyk, Annika L Gable, David Lyon, Alexander Junge, Stefan Wyder, Jaime Huerta-Cepas, Milan Simonovic, Nadezhda T Doncheva, John H Morris, Peer Bork, et al. String v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic acids research*, 47(D1):D607–D613, 2019.