**6/10/2019**

Dataset gene\_disease.tsv

* There are 1,137,270 instances with selected columns where selected columns are the following: geneId, diseaseId, and diseaseClass.

1,137,270 instances

Number of unique value of each columns are as followed

geneId = 16629

diseaseId = 9464

diseaseClass = 815

pmid = 398198

source = 14

**Selected Classes of gene\_disease.tsv is the following, rank by frequency in descending order.**

label = 0050686, count = 157 "organ system cancer"

label = 16, count = 87/115 "integumentary system disease"

label = 3093, count = 75/47 "nervous system cancer"

label = 0080001, count = 67 "bone disease"

label = 863, count = 66 "nervous system disease"

label = 77, count = 65 "gastrointestinal system disease"

label = 0050155, count = 64 "sensory system disease"

label = 0050117, count = 58 "disease by infectious agent"

label = 66, count = 57 "muscle tissue disease"

label = 0080006, count = 50 "bone development disease"

**new\_mega\_classes (highest possible label classes)**

0050117, disease by infectious agent

7, disease of anatomical entity

14566, disease of cellular proliferation

150, disease of mental health

0014667, disease of metabolism

0080015, physical disorder

225, syndrome

**Dataset gene\_disease\_50000\_label\_no\_None.txt**

Dataset of the first 50000 qualified instances of gene\_disease.tsv where all instances belong to 1 of the selected mega classes mention above.

50000 = the first 50000 qualified instances of gene\_disease.tsv.

Qualified instances are instances that have all the selected features: geneId, diseaseId, and other features. Other possible features are diseaseType, diseaseClass, diseaseSemantic Type,and score. Meaning of these features are not clear to me. So,as of now, I decide not to use any features. Many papers do not use additional feature to do node classification.

Label = label column is added at the right of the dataset

It is a bipartite dataset where gene only connects to disease and disease only connects to genes. I am not sure if I should use diseaseClass as a feature when we are trying to do label nodes by classes. DiseaseClass has the following format

no\_None = there is no None class. None class is class that is not in the selected classes.

* Out of the first 50000 stances in the gene\_disease, there are 1995 instances
* Training and testing are separated randomly with no further criteria.
  + In full network where all nodes are included. Number of edges is a lot more than number of uniq\_nodes. However, in the small network, such as data with 50,000 instances. I am not sure if there should have any criteria to choose which nodes should be included in the training and testing.
  + I do not think there should be any criteria other than splitting them randomly and maintain the amount of number instances ratio per label. This is because degree of nodes in 2 bipartite groups hold necessary information for building GCN.

Comments:

+++++++++++++++++

Type of nodes:

1. Disease node
2. Gene ID node
3. Verify that each diease node is mapped to one class lable or there is a possibility to map to multiple disease lables
4. Check whether it is possible to lable Gene nodes, and how?
5. Currently, we have COPD related diease: diseaseId = 9464. This is rather too large, considering human disease network. Please double check the connections are correct.
6. Let’s focus on build the complete network (COPD) using all instances, then we can reduce the network, if it’s too computational expensive to learning the network. (check code to save)
7. Let’s plan to publish COPD network by June 22 through project website (also report to NSF).

+++++++++++++++++++++++++++++++++

**6/14/2019**

disease\_gene.tsv is used to collect disease label as well as disease’s instances.

Disease label/classes are commorbidiy of copd as shown in picture below.

Number of classes is 9.

doid\_label\_dict = {'3393': 'coronary artery disease'**,** '2841': 'Asthma'**,** '9351': 'diabetes mellitus'**,** '0060224': 'atrial fibrillation'**,** '6000': 'congestive heart failure'**,** '10763': 'hypertension'**,** '1168': 'familial hyperlipidemia'**,** '1596': 'mental depression'**,** '6713': 'cerebrovascular disease'}

Step to generate disease Instances with selected classes:

* I manually selected it to only includes the most relevant comorbidities by adding 1 class at time until number of instances is large enough (in this case 11657 instances)

Note: there are 2 ways to increase number of instances

1. Increase number of classes
2. To increase number of instances without increases number of classes, I must substitute old classes with a new classes that contains more classes.

Number of all the instances

{geneId: 11657, geneSymbol: 11657, diseaseId: 11657, diseaseName: 11657, diseaseClass: 11657,pmid: 11657, source: 11657, class: 11657}

Number of uniq value in each classes

{geneId: 2429, geneSymbol: 2429, diseaseId: 61, disesaeName: 61, disesaeClass: 19, pmid: 6973, source: 11, class: 9}

Difficulties

I manually select labels by referring to multiple copd commorbities papers and search for the most relevant copd comorbidities that are gender and age adjusted.

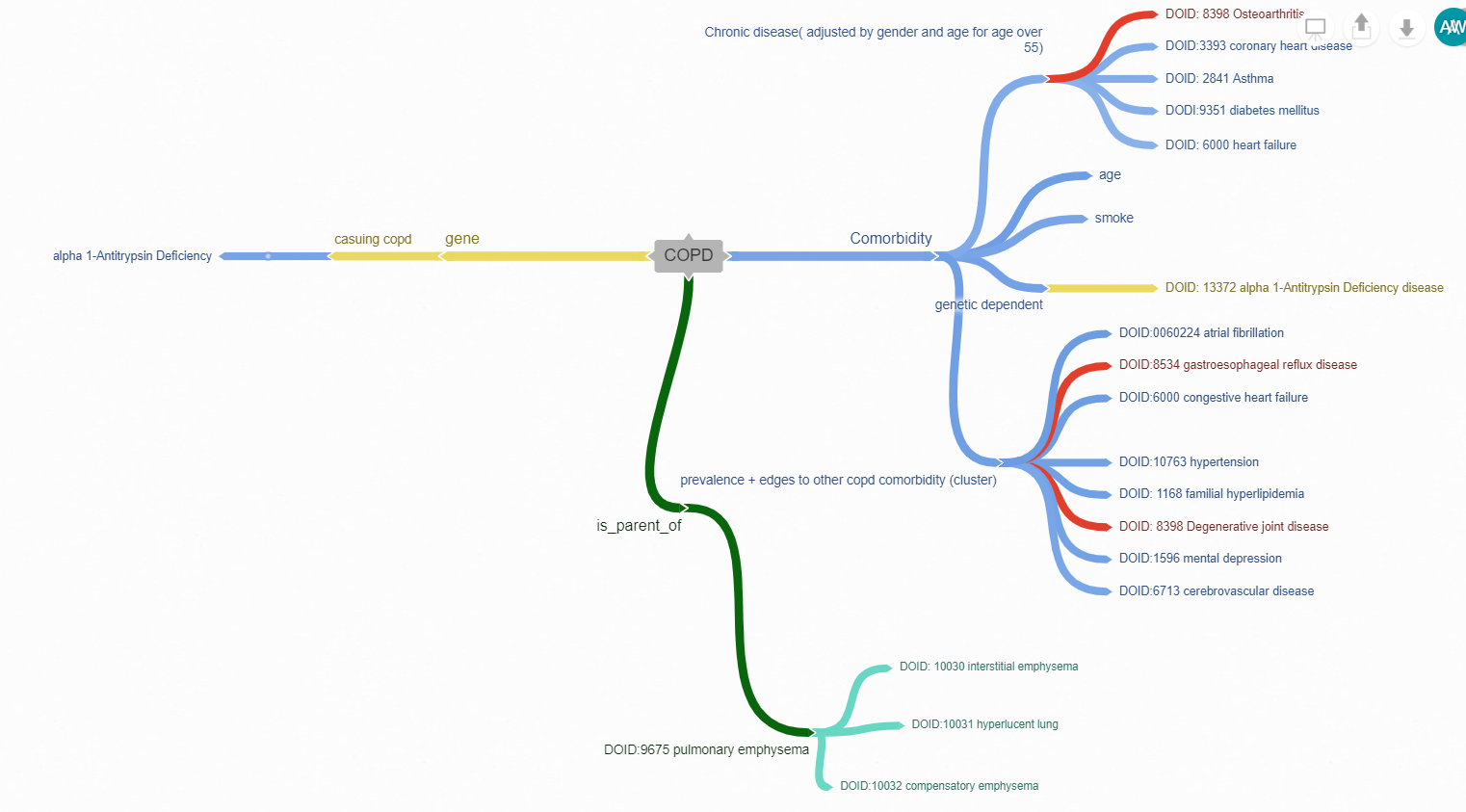
However, copd is commonly found in smoker and older population, so age adjusted that I could find is average of 40 and 55. (on 2 different papers)

I must use copd commorbidites of smoker because 90 + percent of copd patients are older and I cannot find copd comobidity statistic that is specific to younger population who are most likely to be non smoker (due to his/her condition). I do realise that this is more beneficial to our gene-disease analysis case, but no useful infomation can be found.

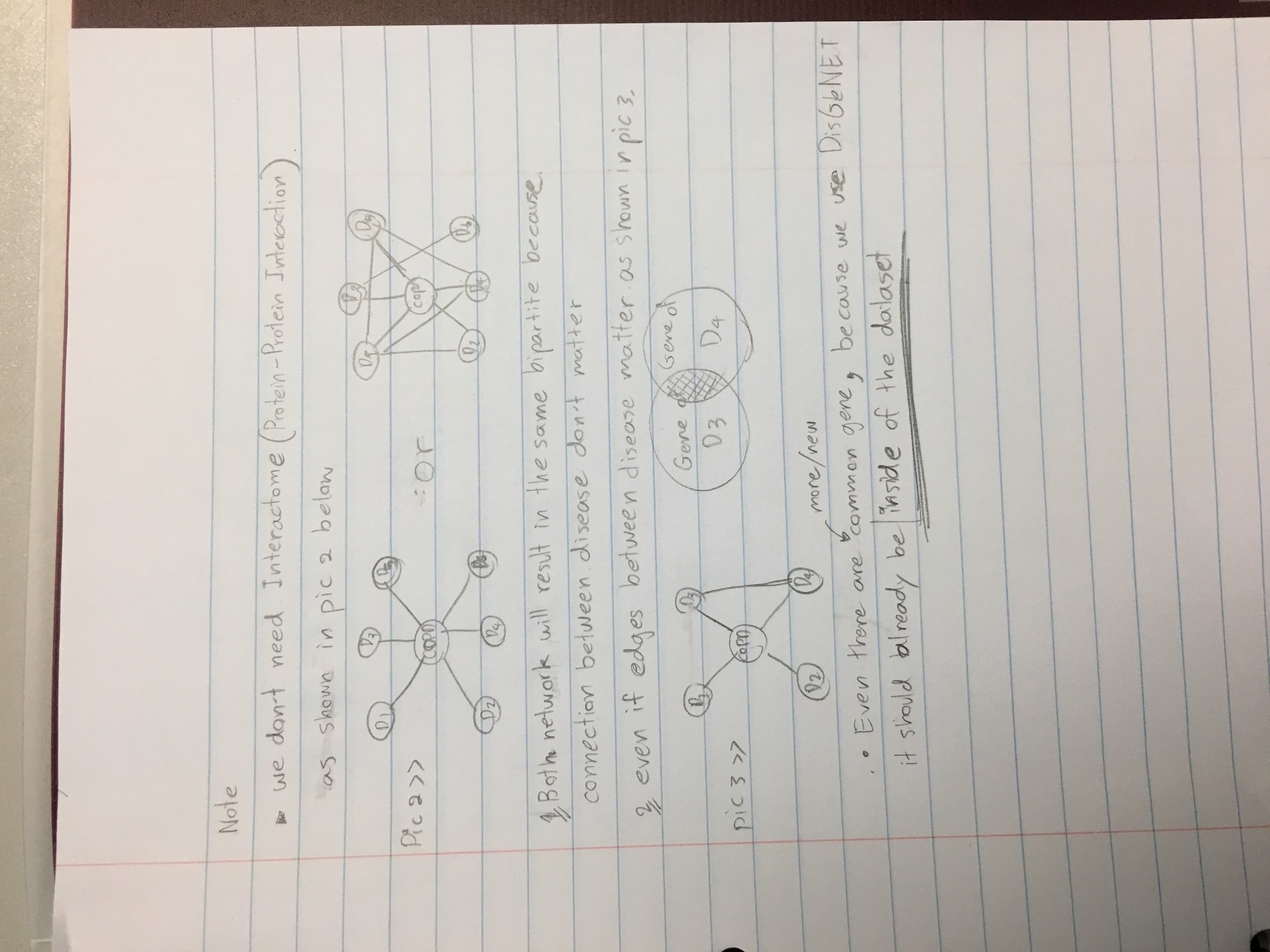
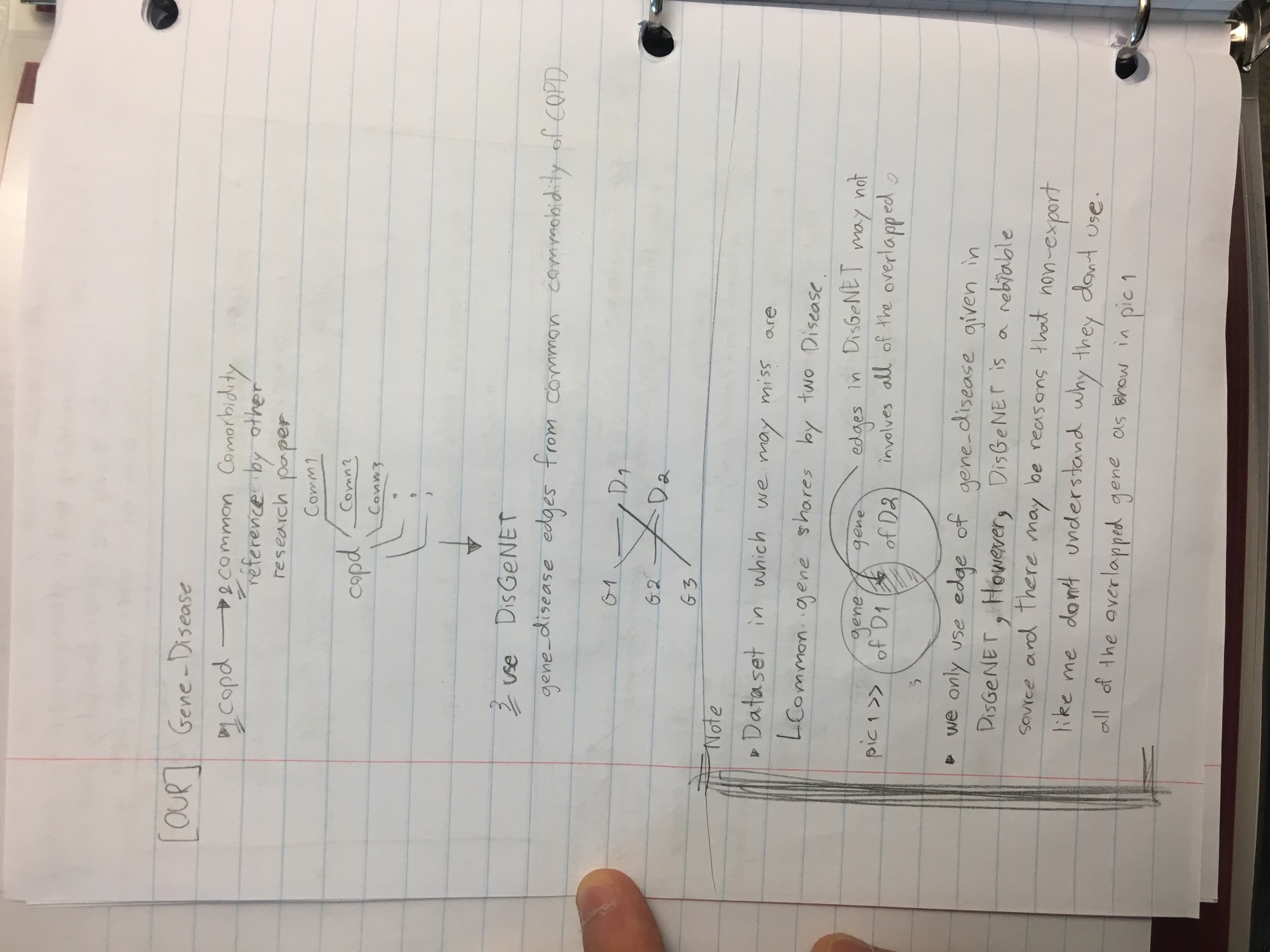
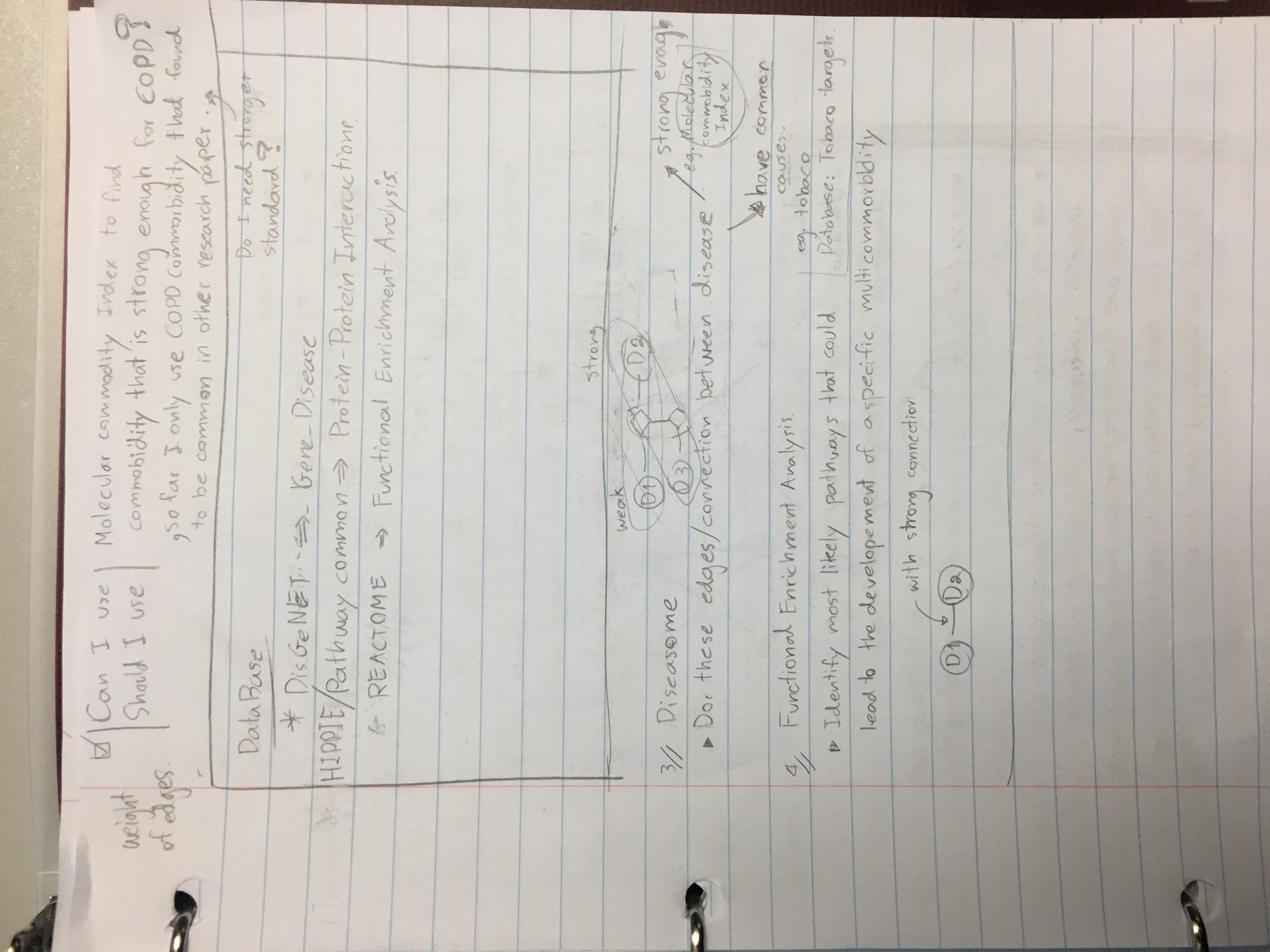
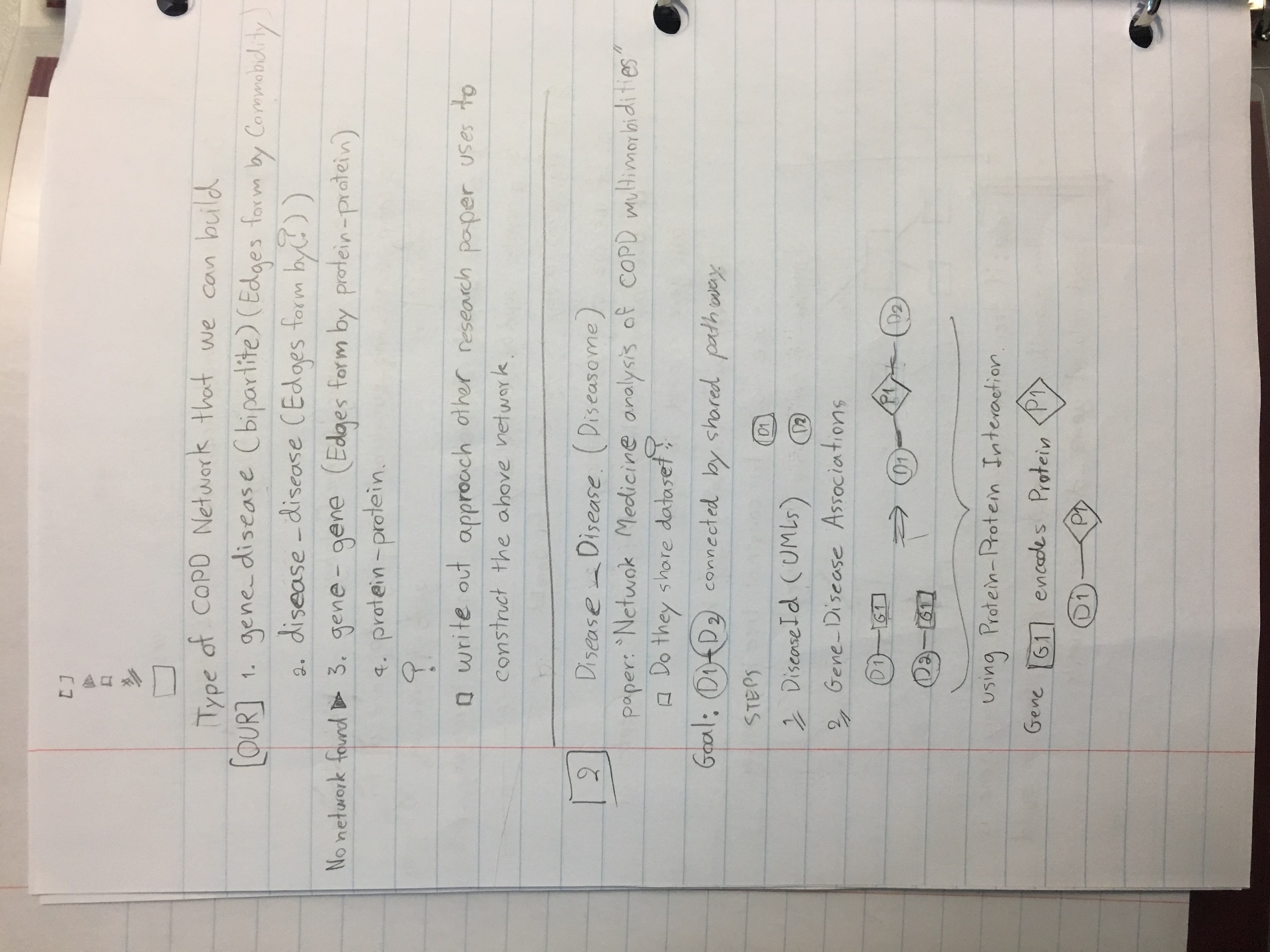
Though, I did find that there is 1 copd gene which is a cause of a cause of "DOID: 13372" or doid of copd, but it is not includes in any of the classes given above, and its doid is a leaf. (no children, cannot be used as label)

for gene, I will link diseaseId to its corresponding gene in disease\_gene.tsv file and then label gene with gene Ontology.

In addition, if possible I will include "alpha 1-Antitrypsin Deficiency" which is a direct cause of copd. However, my concern is that there may be no edges connect to any other nodes in disease because as i mention above doid of copd that is cause my "alpha 1-Antitrypsin Deficiency" is not included in any of the selected classes.



Summarize and comparing our approach to other approaches use in other research papers



**6/17/2019**

* Created unique node and edges in excel
* Created html website to describe our dataset.
* Implemented Graph AttentionWalk with pytorch using chameleon dataset
* Implemented Deep Walk with pytorch using Cora dataset
* Implemented GraphConvolutional Neural Network with pytorch Cora dataset
* Implementing Bipartite Network Embedding (half a day of work)
* Plan to implement GraphSage (half a day of work)
* Experimental structure
  + 80/20 splits on training and test
  + Baseline of embedding algorithm is no embedding
  + Embedding Algorithm that will be used
    - Graph AttentionWalk
    - Node2Vec or DeepWalk
    - BINE
  + Node classification Algorithm
    - GCN
    - GAT (Graph Attention Neural network)
      * How will AttentionWalk and GAT work together?
    - CapsuleGCN (because it claims that capsule can capsulr
      * It cliams that Its “capsule” capsule features of the graph.
      * how will these feature work with feature of embedding Algorithms.?
      * This may require me to manually code the algorithms, because it is not conventional and widely used
    - GraphSage
      * It is a substitution of CapsuleGCN, in case it takes to long to implement.
  + I will use pytorch geometric for algorithms that are available.
  + Visualization
    - Tsne for node classification
    - Loss functions
  + Measurement will include the following
    - Confusion matrix
    - Overall accuracy
  + I am not sure how to split disease and gene nodes.

June 24 2019.

Comments:

1. Change the html file format (simplify):
   1. degree distribution plot (use gene degree distribution plot and disease degree distribution plot).
   2. Draw diagram to show the input dataset (or data sources, such as gene ontology API) so users can understand was the dataset generated
   3. Put nodes in one file (including disease node label).
2. Cross validation (“am not sure how to split disease and gene nodes”)
   1. All genes-diease nodes are kept in the networks. For traingin, split diease nodes as 80% labled, and use them to predict remaining 20% nodes.
3. Change “COPD bipartite graph info” as “COPD Gene-Disease Network”

6/24/2019

1. Make all of the selected model compatible with copd\_label.txt dataset, but 3
   1. CapsGCN
   2. BINE
   3. GraphSage

7/1/2019

* Obtain all embeddings values from BINE, AttentionWalk, and Node2Vec
* Finished building Dataset inherit from pytorch\_geometric.data.Dataset abstract class easily be fed into GCN, GraphSAGE, GAT, and CapsuleGCN.
* (Currently doing) I need to run the files to get result.

7/15/2019

* Highest possible accuracy obtain is 87% which is extreme outlier with a specific hyperparameter configuration and not generalized when test and train are randomly splitting

Comment: when the dataset is imblanced (one class of samples are significantly more than other classes. The dataset is imbalanced. In this case, classifiers may tend to classify instances into the majority class. So accuracy is not a good measure. You may consider other measure, e.g., AUC, F1-score etc., in combination with the Accuracy).

* With random seed, model cannot generalize well.

Comment: This is a typical problem. If a model simply classifier all instances into one class, it cannot generalize well.

* Out of the 5 classes, 1 class is significantly less than others (8 compare to the others class members of range from 20-30), so I used to have also tried using weighted loss function which helped improve the model slightly.
* With unsupervised node embedding, gene seems to have distinct cluster, thought I am not sure if it is informative fore node classification.

My plan to improve model prediction

1. Supervised Node embedding vs unsupervised node embedding
   1. Supervised node embedding helps model discriminate between classes.

Comment: This is true. Supervised node embedding is helpful to find better features to differentiate nodes between different classes.

1. Increase GCN depth to Gain more info from graph structure
   1. Prevent over fitting with residual network

Comment: Do you mean random walk “depth”? In a bi-partite network setting, random walk depth may not be very useful.

1. Feed pseudo label in more instructive way.
   1. Only choose the top most informative pseudo label to be calculated in loss function.
   2. It is shown that streaming is better than pooling active learning.

Comment: Item 3 can be a possible solution. This is a solution to solve unlabeled data point challenges. This is also called semi-supervised learning. You can try to implement a solution (following Item 3), and compare its performance to baselines (e.g., methods without using pseudo lables).

1. Reformulate problem to multi\_classes?
   1. So we do not have to deal with unlabeled problem.

Comment: I am not sure about item 4. Even if we have multiple classes, we still have unlabeled data points. Our goal is to assume that some nodes in the network are labeled, and we are trying to predict lables of unlabeled nodes.

7/22/2019

* Using pseudo label, I have successfully increase overall performance to 1 misclassification which account to 96.5% accuracy.
  + Unlabeled class is ignored. (aka class weight = 0)
  + Also, I have noticed something interesting, and it may be worth to write about. There could possibly be false in my experimental setup, though I failed to find them in the present.
  + In streaming and pooling topk pseudo label, pseudo labels are only assigned to unlabeled nodes.
  + However, using the mentioned approaches, **it causes overfitting to a class with majority instances and does not improve performance in any standard** F1, precision, recall, and accuracy. Varying cost of each class does not help overfitting to one class.
* I successfully increase overall performance using the following configuration.
  + GCN without node embedding as features
    - Node2vec showed no improvement and it also takes more iteration to reach maximum accuracy.
    - I have not tried this approach with GAT, GraphSAG, bine, and attentionwalk because performances of these models showed that they cannot learn anything.
  + Unlabeled class is ignored. (aka class weight = 0)
  + I allowed topk to be picked from labeled instances without replacement.
    - **labeled instances without replacement implies that labeled that is already assigned pseudo labeled can be picked again with possibly different predicted labels**. As a result, at the very first iteration, topk may be picked from unlabeled instances, but as iteration increases, Topk will only be picked from labeled instances. This approach only misclassified 1 instance during test set with randomized test dataset.
    - Observation: Number of same nodes being included in calculation of loss function is proportional to weighted being assigned to that node?
      * Weighted node is used instead of weighed class?
  + In Topk with labeled instances,
    - I found experimentally that using topk=0 (pseudo label is disable) has the expected performance as training without pseudo label.
    - topk = 1 is enough to obtains highest performance, though with higher number of iteration than optimum topk.
    - Topk = 2 is enough to obtains highest performance with lowest number of iterations.
    - In addition, with my dataset, I found that topk = 20 is achieved highest accuracy and lowest iteration most often. This needs further validation for different dataset and set up.
  + I have also tried “topk with replacement.”
    - In topk with replacement, in epoch i where i =1,2,3,…,n where n = maximum number of epoch, topk labeled is picked and be calculated with pseudo label in which they are assigned according to model prediction.
    - Using the mentioned setup does not improve performance in all measurements and lead to overfitting as I have expected.
* Using labeled instances without replacement, it is fair to say that disease uses genes as features using graph structure rather than nodes context.
  + Since genes has no connection between itself, it can be interpreted as features with no observable correlation or relation.
  + I failed to find similar literatures that learn features as graph structure instead of node context.

Question:

1. Is there possibly false in my experimental setup?
2. If answer to question 1 is no, should I process to construct experiment that implement features as graph structure instead of node’s context in other environmental set up such as dataset and other GNN technique?
3. Should I add features to gene and disease? so observation can be written in relation to medical finding.
   1. I have already listed features to be used for both gene and disease, but I have not yet process on adding these feature
4. Should I focus on hyper parameter tuning GAT, GraphSAGE, bine and attentionwalk first?

**8/29/2019**

* I have found an error on my code on topk pseudo labeled.
* I have also fully investigated the possibility of using features as node and learn by propagation.
  + I come to conclusion that the approach has fundamental flaw which need to be addressed as followed
    - Adding more edges will ‘diffuse’ information of node’s context during propagation. This can be fixed by fixing number of edges to propagate information as seen in “feature convolution layer” in GCN.
    - GCN does not learn weight of edges (explained the reason of weight of edges below)
* Better approaches would be the following. I have not yet tried to implement any of the options
  + Predict connected edges between nodes that have similar node context though are not directly connected in the graph (assuming that there may exist predictive properties of nodes with similar context) to avoid 1. diffusion of node’s information 2. Convergence speed.
    - However, using link prediction, the model doesn’t allow for assumption that some features have strong/weak correlation to nodes of a given class.
  + Create 2 networks: node to features (in bipartite network) and node to node as in ordinary graph, and impose disagreement constraint (such as entropy regularization) so the two graph can learn together.
    - In Node to features network, model must focus on learning weight of edges instead of node context. (given a node has value =1, any number can be produced by multiplying node’s value by weight of edges) Node to features network allows the network to learn features embedding of nodes with no graph structure given.
    - Node to Node network allows the network to learn nodes features embedding of node with given graph structure.
    - This is similar to saying “There exist combination of features that are predictable for any given class”. It is the same way that one can see differences in picture of cat vs dog.
    - Intuitively, this approach would allow the model to learn intrinsic properties of any class, and this should be helpful predicting unseen dataset.
* Adding edges feature to bipartite graph
  + I have added DSI and DPI as edges features which I haven’t yet obtained the result. (problem with DSI and DPI is specified in “Dataset Problem” section )
* Dataset Problem.
  + Problems lie in structure of bipartite network.
  + Given gene\_disease dataset from DisGENET, I can create diseases and genes bipartite graph of copd.
  + The problem here is that not all genes have direct relationship to certain diseases for the following reason. (unless it is a genetic disease)
    - Genes + external influence cause certain diseases. (without data from patient, dataset cannot capture this properties)
    - A Gene has variant. Not all variant can cause certain diseases. (Do we consider this gene to be a cause of the diseases?)
    - Diseases are caused by combination of many genes. (without gene to gene connection, bipartite network cannot capture this properties)
  + Dataset from DisGENET follows certain standards. (In API, there is an option for p-value of connected between genes and diseases)
    - This prevent me from adding gene to certain diseases as well as adding features to certain gene.
      * I cannot find “absolute” measurement as features of genes. I only found relative measurement between genes (using patient data, gene expression) or between diseases. (using disease pathway)
      * I found gene expression to
  + There aren’t measurements (I cannot find one so far) that indicate intensity of connection between disease and gene.
    - DSI and DPI values can only be used with MeSH standard. Disease Ontology cannot use DSI or DPI because DSI and DPI values are related to assigned disease classes of a given disease (1 disease could belong to more than 1 classes)
  + To Conclude the problem, with bipartite graph, I can only use data from
* Proposed solution to dataset problem.
  + Reframe problem into multi label problem this may help improve performance of pseudo label.
    - with single class assigned to disease, gene cannot have labels. Pseudo label must assign a class to gene. This does not make much sense consider that naturally gene belongs to multiple classes
  + adds protein deficit as gene’s features (if gene produces the deficit protein).
    - Protein deficit have direct relationship with certain diseases.
  + Phenotype, Gene expression has possibility of being used as gene features
  + Symptoms, Organs involved have possibility of being used disease features.

Questions

Due to difficulty of Dataset Problem given above, is it possible to collaborate with student/professor from bio department? If it is what is the best way to contact them?

8/26/2019

What I have done

* I have created all intended models to be tested.
  + Implemented baseline Logistic regression
  + GAT, GAT, GCN,node2vec, attentionwalk, BINE
  + Implemented Variation of Loss functions
  + Implemented variations of Pseudo labels.
  + Implemented Graph creation that support all the test combination scenarios.
  + implemented both random and grid hyper-parameter search.
* I have also implemented my own algorithms which failed to perform due to fundamental flaws of the algorithms.
* I run hyperparameter search for gcn, gat and

Results

Gcn\_no\_feat

Gcn\_bine

Gcn\_attention\_walk

Gcn\_node2vec

Gat\_no\_feat

Gat\_attention\_walk

Gat\_bine

Gat\_node2vec