Tom Evans, Kavin Aravind, Rishi Pulluri
Team Wildcards
Due: May 2nd, 11:59PM

CS 4984: Computing the Brain

Mouse Connectome

Goal

Analyze the structural properties of a comprehensive connectome of the mouse brain, collected from findings in over 185 publications that appeared in literature since 1974. If time permits, correlate this network with the mouse connectome created independently by the Allen Institute for Brain Science

---

Research Paper Overview

Organizing Principles for the Cerebral Cortex Network of Commissural and Associative Connections

Definitions:

Commissural: A band of nerve tissue connecting the hemispheres of the brain (between sides)
Cognition: The mental action or process of acquiring knowledge and understanding through thought, experience, and the senses
Monosynaptic Association: Within one side

This study investigates the organization of cortical association and commissural connections in the rat for which the greatest amount of relevant structural data are available.

Dataset S1: (pnas.1712928114.sd01.pdf)

All relevant data in the primary literature were interpreted in the only available standard, hierarchically organized, annotated parcellation and nomenclature for the rat brain.

Dataset S2: (pnas.1712928114.sd02.xlsx)

Association and commissural connection reports were assigned ranked qualitative connection weights based on pathway tracing methodology, injection site location and extent, and described anatomical density. This dataset contains all ceollated connection report data and annotations.

Dataset S3: (pnas.1712928114.sd03.xlsx)

The data extracted from these reports (Dataset S2) to construct connection matrices are provided in this dataset.

Tom Evans, Kavin Aravind, Rishi Pulluri
Team Wildcards
Due: May 2nd, 11:59PM

Introduction

Our presentation both in class as well as VTURCS focused heavily on Overlapping Modules, so we decided to run the CONGA Algorithm (Gregory, 2007) which extends Girvan and Newman's well-known algorithm based on the betweenness centrality measure, the CFinder Algorithm (Palla et al., 2005), which analyzes cliques of varied size in a network, and the COPRA Algorithm (Gregory, 2010), which extends on the label propagation technique to assign module identifiers to nodes in a network on the Mouse Connectome. We also worked on a variety of visualization techniques and plotted various network metrics from the mouse connectome data.

One thing in particular to note about this report is that the terms 'mouse' and 'rat' are used synonymously. Although these species are closely related with slight differences in brain structure, both terms were used because the project spec. referred to the data as a mouse connectome, but the paper revealed that the network was representative of the rat brain.

Implementation

The majority of the code for this project was written in Python due to its ease of use and wide availability of graph libraries; however, there was also experimentation with Java's GraphStream library and Golang for file processing. The main Python graph libraries used were *python-igraph* and *networkx*, and the main visualization libraries were *plot.ly* and *matplotlib*. Throughout the implementation, *python-igraph* was used in conjunction with *plot.ly* to generate interactive three-dimensional renderings of connectomes, and *networkx* was used in conjunction with *matplotlib* to generate two-dimensional graphs and charts.

The first step for implementation was to gather and understand the dataset for the mouse connectome. The challenges associated with acquiring an accurate and comprehensive edge list will be discussed later, but the two main networks used throughout the implementation were the unweighted mouse connectome (308 vertices, 11,858 edges) and the weighted mouse connectome (154 vertices, 5394 edges). The unweighted edge list reflects all *possible* connections in the mouse brain, so nonexistent edges with weight 0 were wrongly included. The weighted edge list reflects only the connections present between all 77 regions in each side of the mouse cerebral cortex. A basic script was implemented in Golang (`csvprocessor.go`) to convert .csv files to .txt edge list files, and additional Python scripts (`file_processor.py`) read the resulting edge lists and loaded connections into in-memory data structures. Another Python script (`process_mouse.py`) was used to write an index which mapped region abbreviations to their full names (`mouse_regions.txt`).

Three-dimensional visualizations were created using the Python bindings for the *plot.ly* graphing library. The code for generating these visualizations is contained in the file `visualize_network.py`. First, the weighted edge list for the mouse connectome was read into

memory to be parsed into three lists: vertices, edges, and weights. Vertices and weights were placed directly into lists, but edges were added to a list of tuples corresponding to the source and destination vertices for each connection. Various graph libraries will generally require elements to be added as either strings or integers, and rarely allow generic typing. In this case, the adjacency matrix was parsed such that vertices were named according to their region and side of the brain. For example, "`one_ECT`" is the ectorhinal area on the left side of the mouse brain and "`two_VISp`" is the primary visual area on the right side of the mouse brain. Since the edge list was written as strings and the graph library required integer indices for vertices, a Python dictionary was used to convert and keep track of vertices in the graph. After generating each list in the appropriate format, a graph object was created from the *python-igraph* library which is a python package built on top of C and C++ (not to be confused with the deprecated *igraph/jgraph* libraries). The graph object in this library allowed for the projection of the graph into three-dimensional space, which translated to coordinates placed in a three-dimensional array. There are a number of force-directed graph drawing algorithms that can be used to represent a network visually, and Swanson et al. used the Fruchterman-Reingold energy minimization layout algorithm to generate figures in the paper; however, our own projection looked horrible when using this layout so we went with the Kamada-Kawai layout algorithm to generate coordinates which looked much better. The *ploy.ly* library contains a large number of 2D and 3D figures, and for this visualization we used a 3D Scatterplot. A figure is traditionally composed of a layout and a number of data traces. Adding a trace for vertices was a simple matter because we already had an array of the x,y,z coordinates, so we could add them to the spatial layout as if they were spherical elements in a normal scatterplot. On the other hand, we could not use the same method to add edges because doing so would cause them to render as spheres identical and indiscernible from vertices. The solution for this was to build a trace for the edges such that each element in the edge list would render as a line beginning at the coordinates for the first vertex (src) of the edge and ending at the coordinates for the second vertex (dst) of the edge.

The `visualize_network.py` script is run with the path to a plain text edge list as input to generate an HTML file that is viewed in the browser. The network rendered in the browser supports a variety of interactive features including hovering your cursor over vertices to view their identities, click-and-drag to rotate, and scroll to zoom in or out. You can view the rendering of the mouse connectome from this implementation by clicking this link.

Jupyter Notebooks was also very useful for this project, as it allowed us to create documents that contain live code and visualizations. In our Jupyter Notebook (Mouse Connectome.ipynb), we initially parse the data from the .csv file that we had already parsed from the two original datasets and then started off by trying to make a visualization of community structure detection without overlapping modules. An online package was used in order to implement louvain's algorithm on our graph dataset and then we displayed it using networkx. We obtained four separate communities using this algorithm and decided to broaden our horizons by finding overlapping communities. We started off again by running k-clique communities with a k value of 0, then we tried to run with a k value of 3, and then a k value of 5 and then 7. We observed the trend of the groups getting more cohesive yet more disjoint and spread out as the value of k was going up. We then tried to plot these values but we had to do a little bit of manipulation to get the data into a format that would be accepted by our graphing software. This was because the results of the dataset were in the format {{node1, node2}, {node3, node2}} where each

subgroup was a community that could contain overlapping nodes. In order to get a visualization where each groups color varied and the overlapping nodes were identified, we had to first get all of the overlapping node from the result, store it somewhere and then separate the rest of the nodes into separate communities based on the group they were a part off while the overlapping would be an all encompassing community that displayed what nodes they were a part off on their label.

In terms of running the CONGA algorithm, we used an online Java JAR file that gave us the output of the each of the overlapping nodes and the number of overlapping communities that they were a part of. We had to convert the data to a file format where we gave them the “to and from connection” for each of the nodes. We also plotted degree centrality, number of average degree, and found the maximum k-clique. We then analyzed the regions that were important in each of these metrics.

Results

Gray Matter Regions

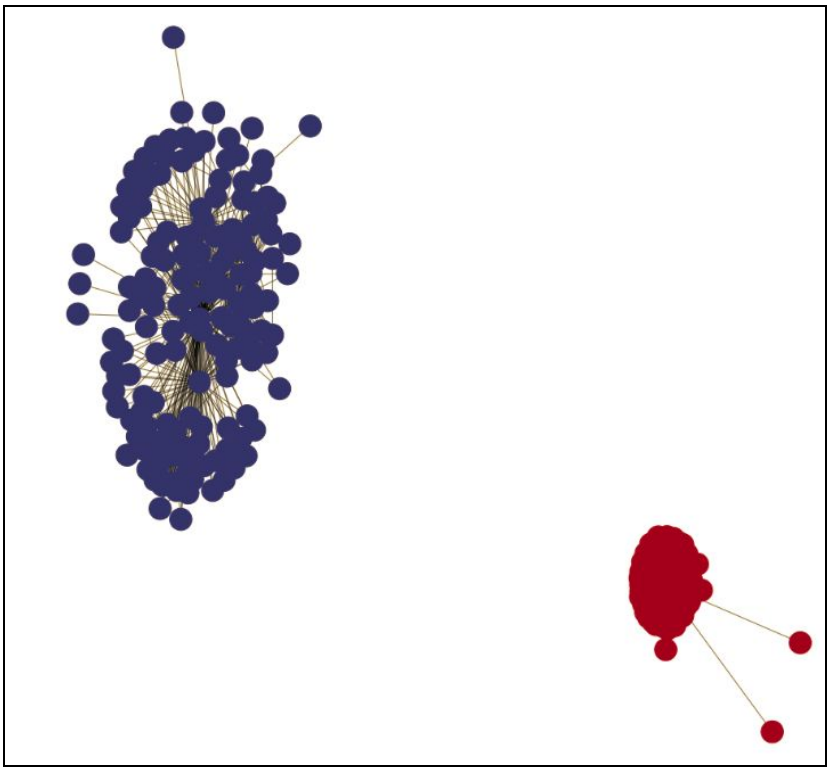
ECT	Ectorhinal area
VISp	Primary visual area
VISal	Anterolateral visual area
TEa	Temporal association areas
VISpm	Posteromedial visual area
SSs	Supplemental somatosensory area
VISam	Anteromedial visual area
VISrl	Rostrolateral visual area
SSp	Primary somatosensory area
AUDv	Ventral auditory areas
6b	"Cortical subplate, Layer 6b"
MOp	Primary somatomotor area
VISlm	Mediolateral visual area
AUDp	Primary auditory area
PTLp	Posterior parietal association areas
AUDpo	Posterior auditory area
VISpl	Posterolateral visual area
AUDd	Dorsal auditory areas
VISli	Intermediolateral visual area
VISll	Laterolateral visual area
VISlla	Anterior laterolateral visual area
ENTl	"Entorhinal area, Lateral part"
ORBm	Medial orbital area
PERI	Perirhinal area
PIR	Piriform area
LA	Lateral amygdalar nucleus
AIp	Posterior agranular insular area
PL	Prelimbic area
BLAp	"Basolateral amygdalar nucleus, Posterior part"
ENTm	"Entorhinal area, Medial part"
EPd	"Endopiriform nucleus, Dorsal part"
ILA	Infralimbic area
BLAa	"Basolateral amygdalar nucleus, Anterior part"
COApm	"Cortical amygdala area, Posterior part, Medial zone"
AIv	Ventral agranular insular area
CA1v	"Field CA1, Ventral part"
VISC	Visceral area
AId	Dorsal agranular insular area
AOA	Anterior olfactory area
TR	Postpiriform transition area
GU	Gustatory area
EPv	"Endopiriform nucleus, Ventral part"
PAA	Piriform-amygdalar area
NLOT	Nucleus of lateral olfactory tract
BMAp	"Basomedial amygdalar nucleus, Posterior part"
SUBv	"Subiculum, Ventral part"
BMAa	"Basomedial amygdalar nucleus, Anterior part"
CA3	Field CA3
COApl	"Cortical amygdala area, Posterior part, Lateral zone"
COAa	"Cortical amygdala area, Anterior part"
NLOT3	"Nucleus of lateral olfactory tract, dorsal cap"
CA1d	"Field CA1, Dorsal part"
TTd	"Tenia tecta, Dorsal part"
SUBd	"Subiculum, Dorsal part"
PA	Posterior amygdalar nucleus
MOB	Main olfactory bulb
TTv	"Tenia tecta, Ventral part"
DG	Dentate gyrus
CA2	Field CA2
IG	Indusium griseum
AOB	Accessory olfactory bulb
FC	Fasciola cinerea
MOs	Secondary somatomotor areas
RSPd	"Retrosplenial region, Dorsal part"
CLA	Claustrum
ACAv	"Anterior cingulate area, Ventral part"
ORBv	Ventral orbital area
RSPv.a	"Retrosplenial region, Ventral part, Zone a"
ACAd	"Anterior cingulate area, Dorsal part"
ORBvl	Ventrolateral orbital area
RSPv.b/c	"Retrosplenial region, Ventral part, Zone b/c"
PAR	Parasubiculum
POST	Postsubiculum
RSPagl	"Retrosplenial region, Lateral agranular part"
PRE	Presubiculum
ORBl	Lateral orbital area
RSPv	"Retrosplenial region, Ventral Part, Anterior Zone"

Here is a list of the 77 regions of grey matter in the mouse connectome. The first column refers to the abbreviations that are described in Dataset S1 and Dataset S2. The second column is the full name of the area.

We were able to create a *networkx* graph using the unweighted mouse cerebral cortex data. You can see the information of the network below:

```
Name: mouse.csv
Type: Graph
Number of nodes: 308
Number of edges: 9255
Average degree:  60.0974
```

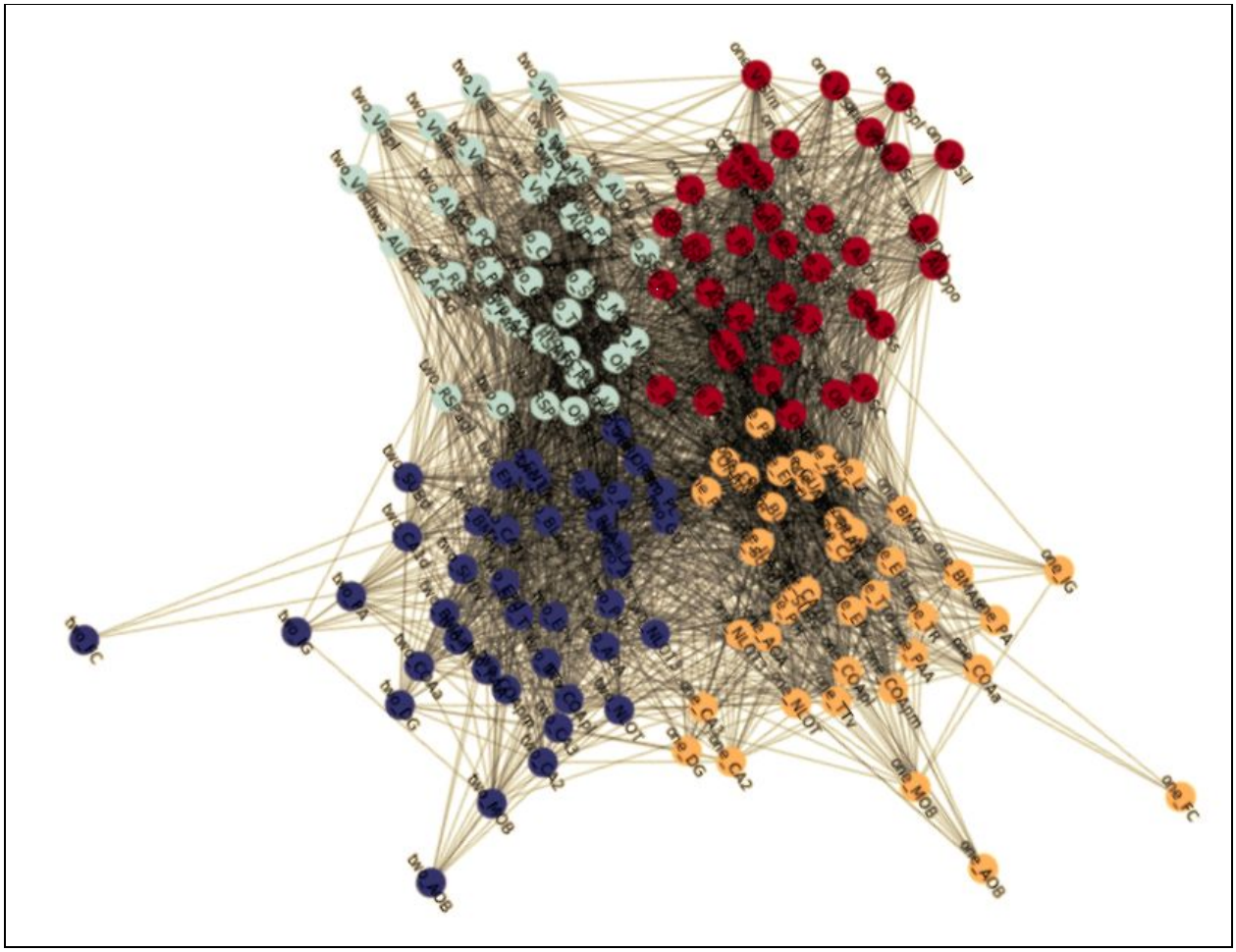
Here is a visualization of this network using *matplotlib*. We did this to get a basic grasp of what we were dealing with. As you can see from the image below, there are two distinct modules, but we cannot determine any more information from this.



We then parsed through Dataset S3, which uses Dataset S2 to create an adjacency matrix for generating a weighted network. Here is the information of this network:

```
Name: final_mouse_weighted.csv
Type: Graph
Number of nodes: 154
Number of edges: 3684
Average degree:  47.8442
```

We then visualized this network with *matplotlib*, which uses the Fruchterman-Reingold energy minimization layout algorithm. Each of the nodes correspond to the regions within the mouse cerebral cortex, while the edges pertain to its connections.



When running Louvain's algorithm for community detection on the weighted mouse connectome, we were able to extract 4 different communities. You can see these communities by the varying colors (dark-blue, light-blue, red, peach). We also were able to label each of the nodes.

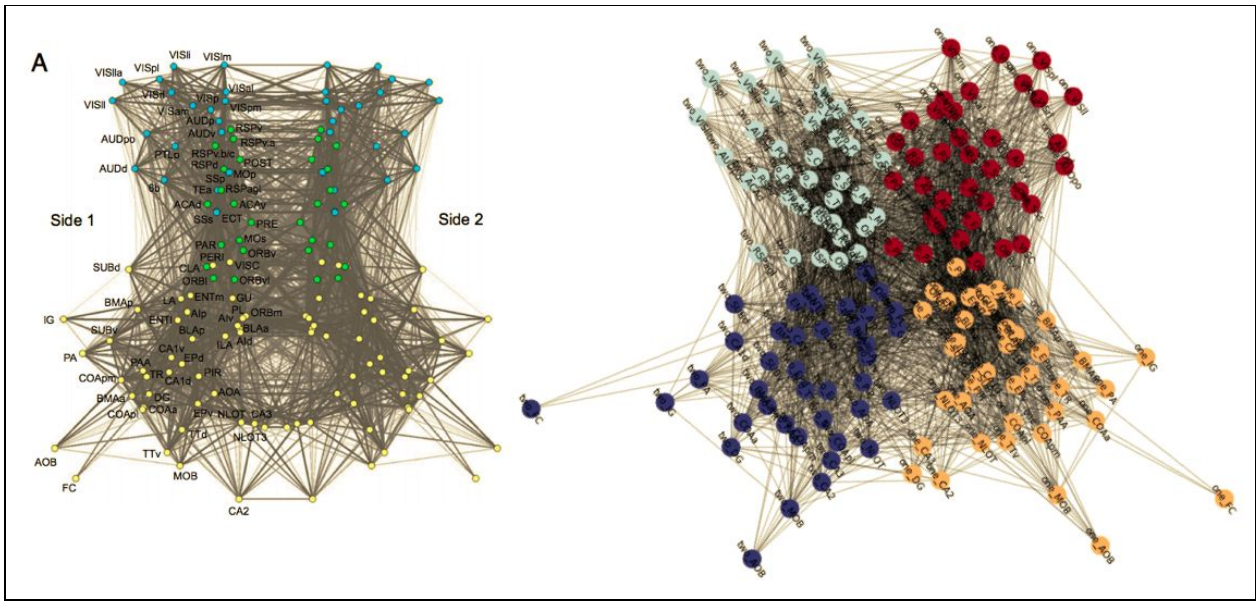
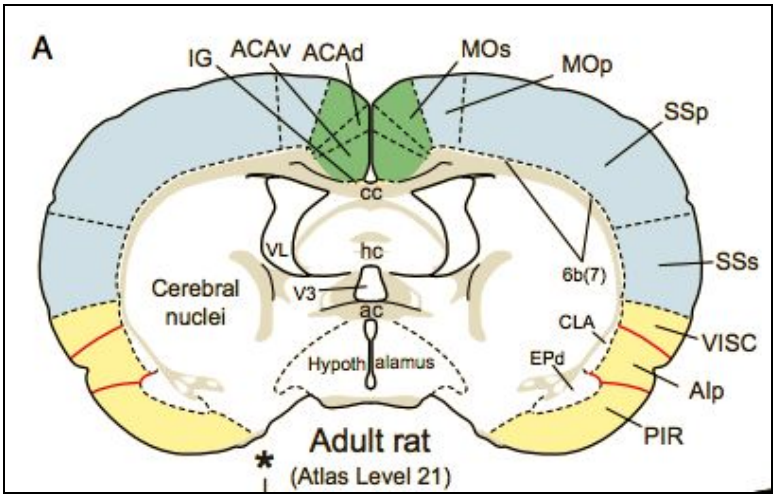


Figure 5A in "*Organizing Principles for the Cerebral Cortex Network of Commissural and Associative Connections"* shows the complete bilateral network of association and commissural connections. This figure is shown above on the left. The nodes and edges are shown two-dimensionally using a Fruchterman-Reingold energy minimization layout algorithm where the node colors are coded by module assignment. The blue vertices correspond to module 1, which is representative of the lateral core. The yellow and green vertices correspond to modules 2 and 3 respectively, which are representative of the ventral and dorsal segments of the shell. In the diagram of the rat's brain below, the blue is the core module with 21 regions, the yellow refers to a shell module with 41 regions, and the green refers to another shell module with 15 regions. We were able to recreate this image almost identically from the paper.

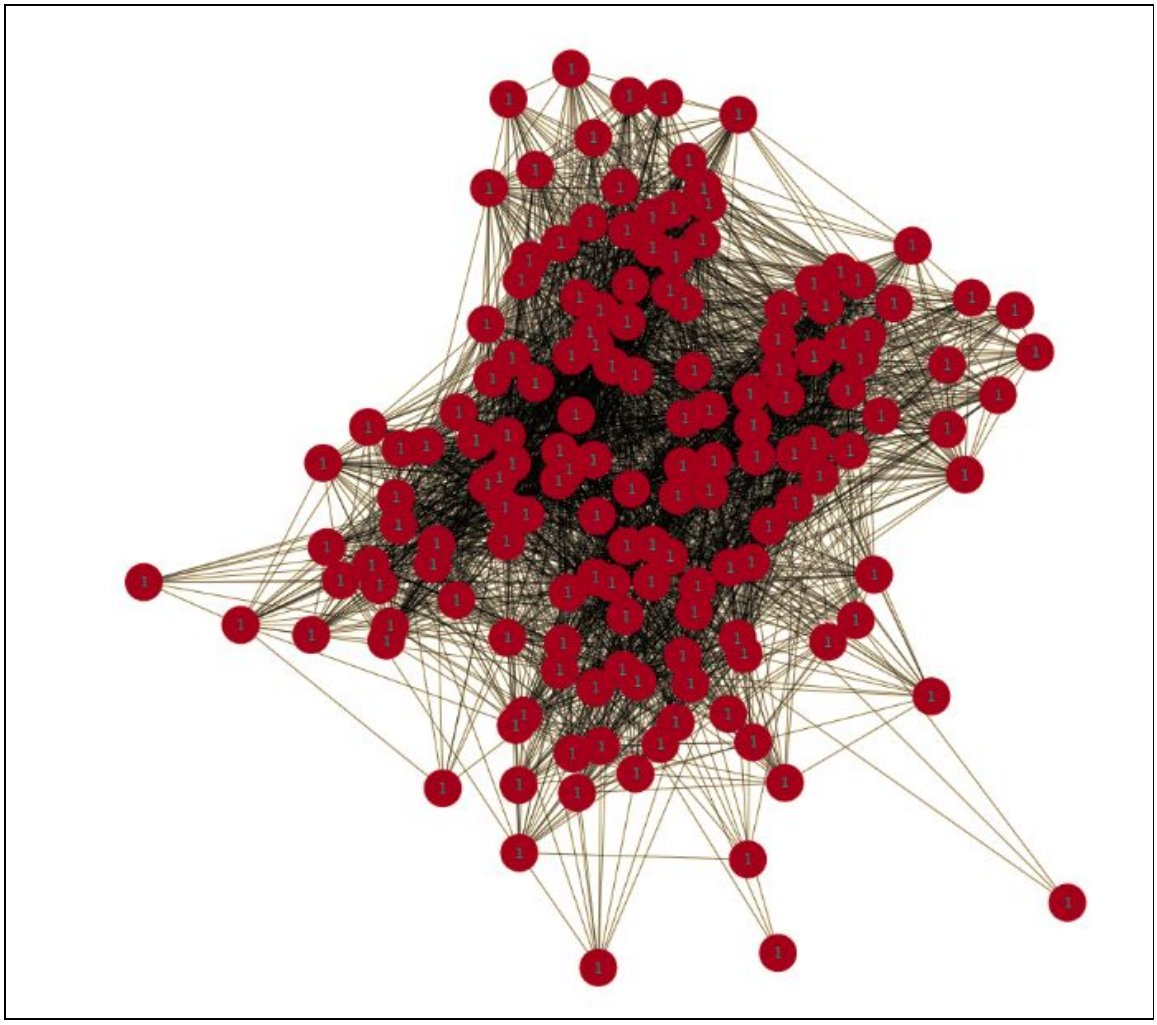
Figure 6A



While the layout of the network in the figure on the top of page 6 appears almost identical to its counterpart on the left, the modular structure is not the same. The figure on the right refers to actual modules found through Louvain's algorithm, whereas the modules on the figure in the left refer to the three different areas shown in the rat's brain (explained in detail above).

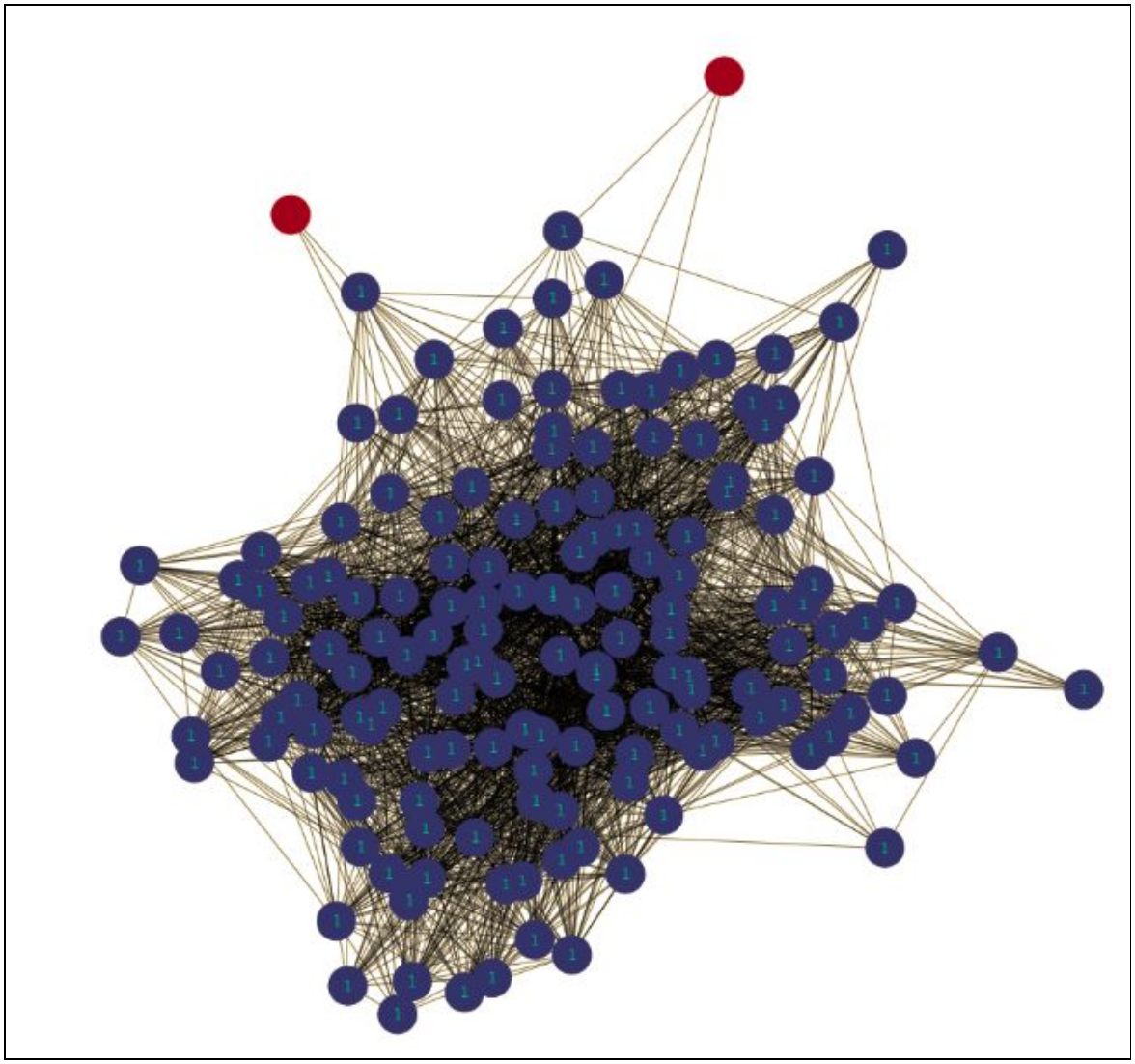
Since our presentation revolved around overlapping modules, we decided to run the algorithms that we discussed on the weighted mouse cerebral cortex data. CFinder was run with varying numbers of K (K = 3, 5, 7).

CFinder (K=3)



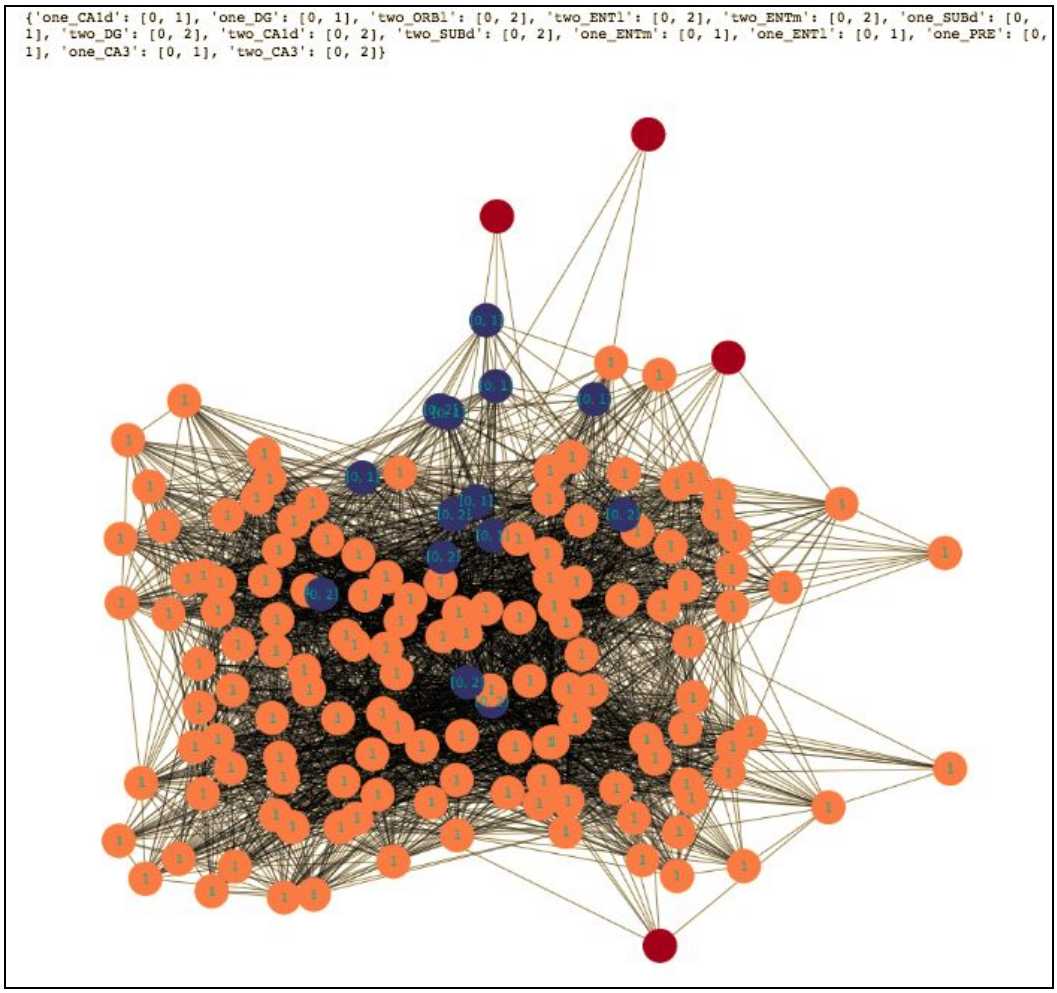
K = 3 resulted in a network where every node was in 1 community, and no nodes overlapped.

CFinder (K=5)



K = 5 also resulted in a network where every node was in 1 community, and no nodes overlapped.

CFinder (K=7)



We determined that using CFinder with a value K of 7 gave the best results. Here you can see two distinct modules where one covers most of the brain while another one seems to group together at the edges. These results did not reveal anything of interest when correlated with descriptions of the brain regions. We also created a dictionary that mapped each of the overlapped nodes to the communities they were overlapping to. This dictionary (key: node, value: community number) is displayed at the top of this image. Each of the communities were assigned an integer value and each of the nodes could be in multiple communities.

CONGA Algorithm

We also tried to run CONGA using the Community Detection Codes Directory. Here is some statistics that we gathered from the unweighted network extracted from the adjacency matrix:

(java -cp conga.jar CONGA conga.txt -e -n 3 -r)

```
================== Statistics ==========================
Initial graph: 154 vertices, 3684 edges
Initial graph: 1 components, with sizes: [154]
Clustering: Final graph size: 1859
Clustering: Vertices split: 1705 total, 148 distinct
Clustering: Vertices split: {one_BLAa=13, one_RSPd=19, two_RSPagl=8, two_VISal=7, two_VISam=13, two_SSs=10, two_EPd=15, two_6b=11, two_BMAa=6, one_BLAp=14,
 two_EPv=7, one_NLOT=6, one_VISrl=8, two_ORBvl=15, two_AUDp=10, two_VISlla=7, two_PERI=20, two_AUDv=11, one_CA1d=6, two_ACAd=10, one_RSPv=31, two_COApm=5,
two_COApl=7, two_CLA=16, one_VISam=13, two_TTv=8, one_VISal=9, one_VISC=9, one_IG=1, two_TEa=18, two_SSp=10, two_MOs=24, two_VISpl=4, one_SUBd=8, two_VISpm
=12, two_AUDd=5, two_MOp=13, two_BMAp=11, one_AOA=13, one_CA1v=12, one_AOB=3, one_ORBvl=16, two_IG=2, one_BMAa=5, one_LA=14, one_AUDd=7, one_BMAp=13, one_T
R=6, two_PAA=4, one_PAA=5, one_VISlm=5, one_ECT=20, two_VISp=11, one_VISli=9, one_VISll=7, one_RSPagl=11, one_DG=3, one_AUDp=11, one_ACAd=15, one_AUDv=10,
one_ILA=17, two_AId=15, two_PTLp=14, one_PIR=13, two_POST=12, one_PERI=29, one_PAR=11, one_AUDpo=10, two_COAa=3, one_PRE=9, two_AIv=10, one_ACAv=12, two_GU
=10, two_ORBm=26, two_PA=4, two_ORBl=11, one_CA3=5, two_ORBv=19, two_PL=19, one_ENTl=19, two_AIp=13, one_ENTm=21, one_CA2=5, one_EPd=11, two_AUDpo=9, two_C
A1v=16, two_CA3=6, two_CA2=4, two_VISC=9, one_ORBm=25, two_SUBd=7, one_ORBl=16, one_POST=11, two_CA1d=7, one_EPv=7, one_ORBv=19, one_NLOT3=13, two_PAR=10,
one_COAa=5, two_ILA=20, one_AId=14, two_ECT=19, one_PTLp=17, one_VISlla=4, two_PIR=15, one_AIp=15, two_NLOT3=12, two_SUBv=9, one_AIv=10, two_VISll=5, two_V
ISlm=7, two_VISli=6, two_PRE=14, one_GU=10, two_MOB=4, one_6b=11, one_MOB=4, one_SUBv=11, two_ENTm=19, two_ENTl=21, two_AOB=1, two_AOA=14, two_BLAa=18, one
_COApl=10, one_COApm=10, one_PA=5, one_TTd=9, one_PL=21, one_VISpm=13, one_TTv=7, one_CLA=20, two_ACAv=13, two_TTd=9, one_VISpl=4, two_NLOT=10, two_RSPv=35
, two_TR=5, two_DG=3, one_VISp=15, two_RSPd=23, one_MOp=16, one_TEa=15, one_SSp=11, two_LA=11, two_VISrl=6, one_MOs=23, two_BLAp=17, one_SSs=10}
Clustering: Betweenness phases: 5389
Clustering: Total time: 77504ms
```

Regions of Interest:

RSPv = 35, **ORBm** = 26, **PERI** = 29

We tried to retrieve the regions from the CONGA algorithm by doing a reverse lookup for the exact name of the communities each of the nodes were a part of, but were unable to do so as the output would not return any results. Due to this, we were not able to discover any conclusive results using this method except for the number of communities each overlapping node is a part of. The interesting nodes to note were RSPv which CONGA put in 35 overlapping modules, ORBm which CONGA put in 26 overlapping modules, and PERI which CONGA put in 29 overlapping modules. We detail the ORBm and PERI regions more in degree centrality, but here is some information regarding RSPv.

Abbreviation	Region Name	Description
RSPv	RetroSplenial region	role in mediating between perceptual and memory functions

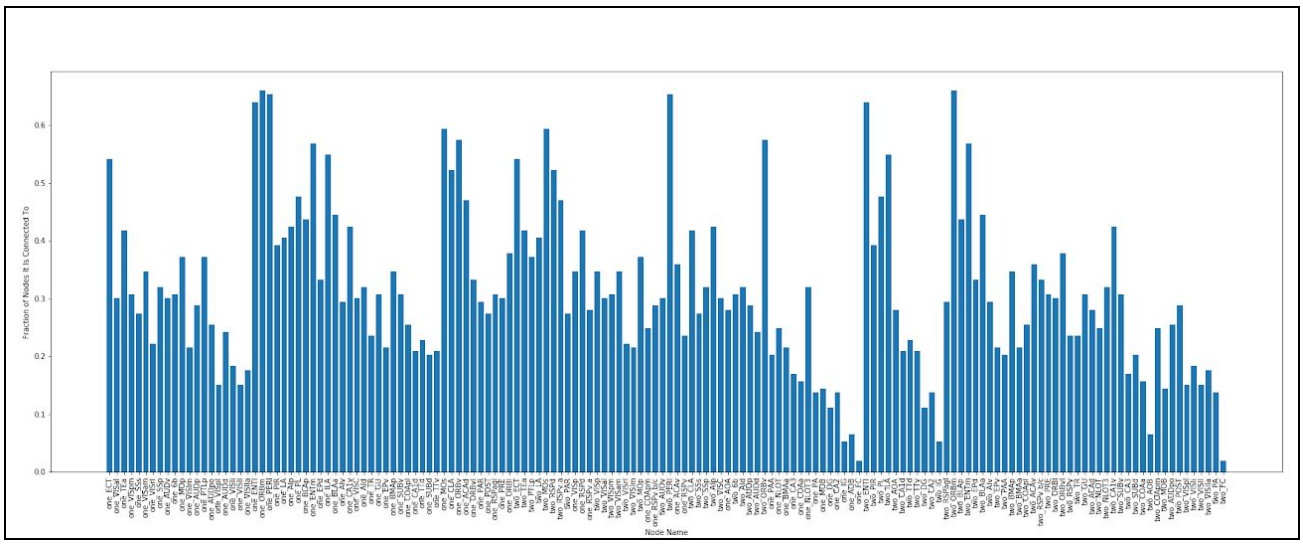
It makes sense that region RSPv was contained in 35 different overlapping modules because it mediates between perceptual and memory functions. The visual data that is taken in through the optical cortex finds its way to this region, while edges going out of this region connect to the region where memory processing takes place. This region seems to be a mediator between both the memory formation region and optical cortex.

COPRA Algorithm

```
************************************
* COPRA v1.25  (c) Steve Gregory 2011 *
************************************
  Network file = conga-weighted.txt
  Network is weighted, unipartite
  154 vertices, 3684 edges
  Weights in conga-weighted.txt range from 1.0 to 7.0
  v = 3.0
  Repeat 10 times and show averages
  Compute modularity wrt conga-weighted.txt
-------------------------------------------------------------------------
1 (1) communities, 13 iterations, 106ms, overlap=1.000, mod=0.000
3 (3) communities, 9 iterations, 44ms, overlap=1.078, mod=0.569
1 (1) communities, 8 iterations, 33ms, overlap=1.000, mod=0.000
2 (2) communities, 8 iterations, 32ms, overlap=1.058, mod=0.623
4 (4) communities, 7 iterations, 32ms, overlap=1.091, mod=0.522
1 (1) communities, 7 iterations, 18ms, overlap=1.000, mod=0.000
2 (2) communities, 11 iterations, 24ms, overlap=1.273, mod=0.511
2 (2) communities, 8 iterations, 18ms, overlap=1.065, mod=0.621
1 (1) communities, 13 iterations, 18ms, overlap=1.000, mod=0.000
1 (1) communities, 6 iterations, 11ms, overlap=1.000, mod=0.000
-------------------------------------------------------------------------
v=3.0
Modularity: best = 0.623, average = 0.285+-0.287
Overlap: 1.056
Communities: 1.800
Non-singleton communities: 1.800
Iterations: 9.000
Total time: 33.600
Termination check time (included in total): 1.600
Simplification time (included in total): 3.300
Vertices: 154.000
Edges: 3684.000
-------------------------------------------------------------------------
```

The output above is the result of running the Java implementation of COPRA on the mouse connectome. We initially implemented the algorithm on our own in Python, but it only worked for unweighted networks. After we realized we needed to use a weighted version of the mouse connectome, the Python implementation became obsolete so we decided to run an existing implementation. From the above analysis it is clear that there is not much overlapping modular structure in this network, as the degree of overlap was only 1.056 when run with an ideal parameter of v = 3.

Degree Centrality



Using *networkx*, we were able to compute the degree centrality for nodes, where the degree centrality for a node n is the fraction of nodes it is connected to and plot them using matplotlib. Although the graph itself is very small, you can pinch to zoom to see each of the abbreviations of the grey matter regions and the fraction of nodes it is connected to. ENTl, ORBm, and PERI were the three areas that had the highest fraction of node connections (~65%). The full names and descriptions of these abbreviations are as follows:

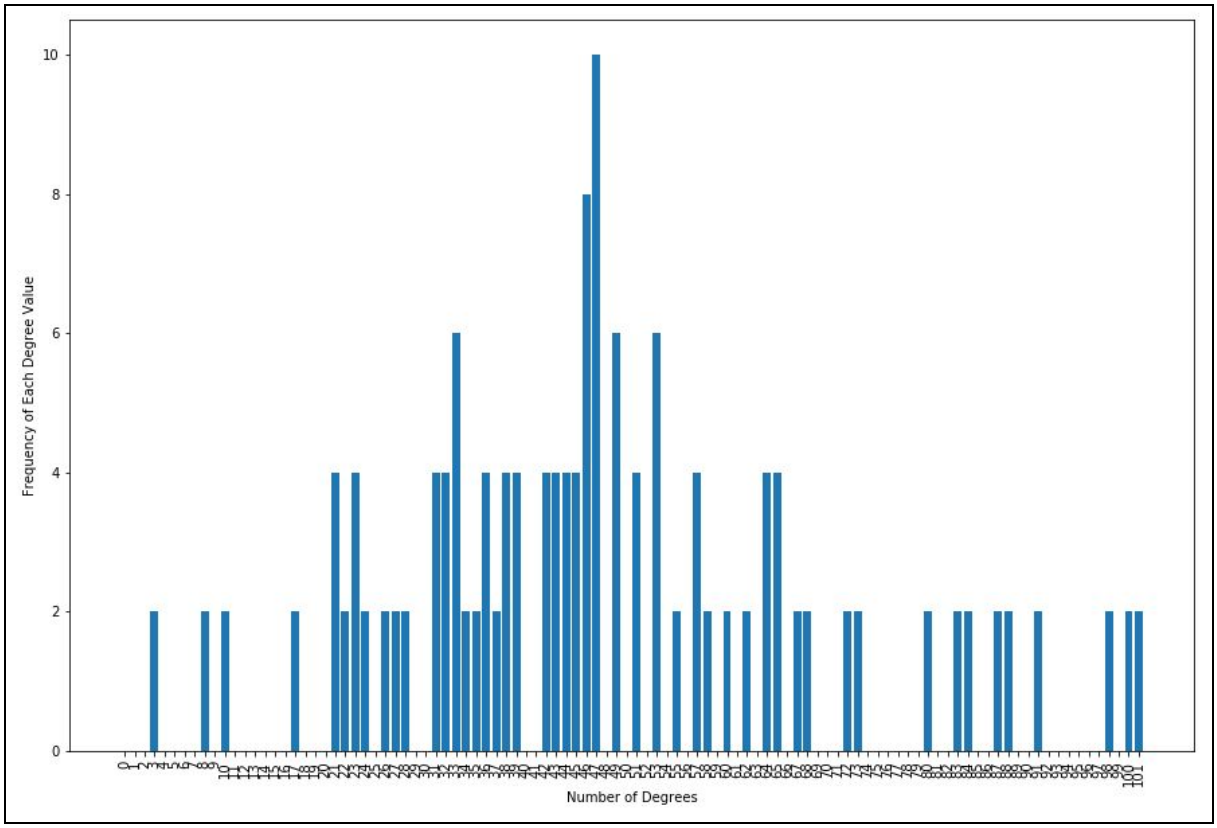
Abbreviation	Region Name	Description
ENTl	Entorhinal area, Lateral part	Hub in a widespread network for memory and navigation
ORBm	Medial orbital area	Area Involved in cognitive processing of decision making.
PERI	Perirhinal area	Area that receives highly processed sensory information from all sensory regions, and is generally accepted to be an important region for memory

Since ENTl is essentially a hub in a widespread network for memory and navigation from our findings, it makes sense that this region would have a fairly high degree centrality as other regions connect to this hub.

ORBm is also a hub in terms of neural connectivity. This is because decision making is a complex phenomenon that occurs mainly in the frontal cortex, which is a much higher in connection density than any other region of the brain.

PERI is also a hub for neural connectivity itself. There is a hub there as well. It mainly connects portions of the temporal lobe to the occipital lobe.

Frequency of Each Degree



The table above shows the frequency of each degree. Since the labels on the x-axis are very small, you can pinch to zoom to see the number of degrees and the frequency of each degree. The minimum number of degrees for any region was 3, while the maximum number of degrees for any region was 101. The maximum and minimum degree count and its corresponding regions are shown in the following table:

Region Abbreviation	Degree	Full Region Name	Description
ORBm	101	Medial orbital area	Area Involved in cognitive processing of decision making.
PERI	100	Perirhinal area	Area that receives highly processed sensory information from all sensory regions, and is generally accepted to be an important region for memory
FC	3	Fasciola cinerea	Primitive band, a part of the cerebellum. Only concerned with reactionary movements.

The FC region is essentially just a band that connects the cerebral cortex to other region of the brain. It is expected that this region would be really low in density overall and that's what we see here as well.

Since ENTI is essentially a hub in a widespread network for memory and navigation from our findings, it makes sense that this region would have a fairly high total degree as other regions connect to this hub.

ORBm is also a hub in terms of neural connectivity. This is because decision making is a complex phenomenon that occurs mainly in the frontal cortex, which has a much higher in connection density than any other region of the brain. This is the major reason why about a 100 connections are made to this region.

Maximum K-Clique in Graph

We also used *networkx* to find the maximum k-clique in the graph, which are the nodes comprised below:

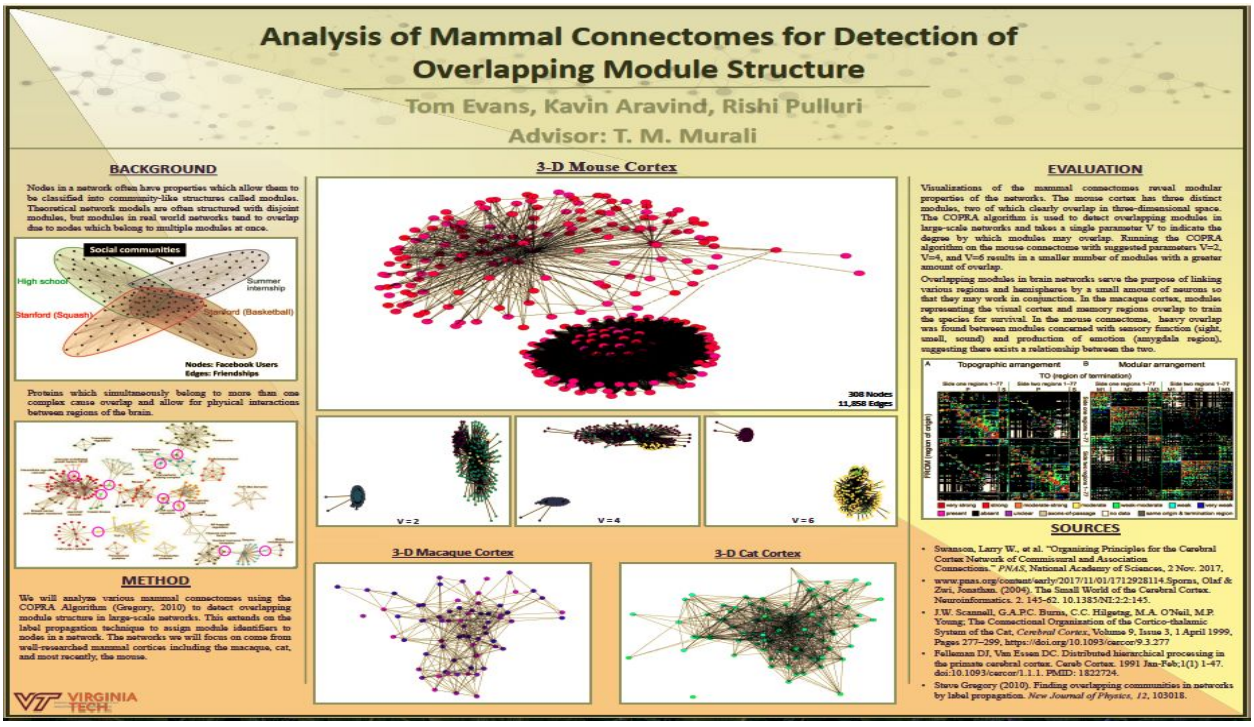
{ 'two_ECT', 'two_ENTl', 'two_ENTm', 'two_PIR', 'one_PL', 'two_PERI', 'two_ILA', 'two_RSPv.b/c', 'two_AIp', 'two_PL', 'two_BLAp', 'two_ORBm', 'two_BLAa', 'two_AIv', 'two_BMAp', 'two_EPd' }

From all of the regions above, it is very noticeable that most of these regions correspond to the perirhinal area and the ectorhinal area within the brain. These two regions are very close together in their location and they also seem to be responsible for memory formation. From our research, it seems that memory formation happens in the ectorhinal region (‘two_ECT’ and ‘two_ENTI’) and memory consolidation tends to happen in the perirhinal region which correspond to ‘two_PERI’ and ‘two_PL’. Even other members of the clique that aren’t as closely related functionality wise are still seen to be related in some way. For example the retrosplenial cortex (‘two_RSPv’) deals in taking past events and trying to convert it into potential action events that the person can execute.

The relationship between the maximal cliques and the relationship between the number of overlapping subregions was also examined. For example, when we looked at our maximal k-cliques and compared it to the the subregions that actually overlapped. There was a tremendous overlap in these two results. There are many different nodes that seem to be in both the maximal cliques and also the overlapping nodes. This probably indicates the fact that more densely a region is connected the probability that region has a lot of overlap in it tends to be high as well. Due to the fact that correlation does not equal causation, we can readily ignore this fact in further analysis.

We wanted to try and test other connectomes and their maximal cliques but their results did not seem to be of much importance because the networks were fairly sparse in general.

Virginia Tech Undergraduate Research in CS (VTURCS)



Our team participated in the VTURCS Spring Symposium where we were able to present our findings both on mouse connectomes as well as overlapping modules. It was a great opportunity to showcase what we had learned throughout this semester. The audience ranged from those who knew very little about neuroscience and graph theory to those who were more advanced. We were able to examine other connectomes that we discussed in class (macaque, cat) which provided more interesting material for this poster. Adding a background column was also very helpful for breaking down the idea of overlapping modules to a real world concept which many people appreciated. Overall we are happy with our work and glad that we were able to present our findings.

Challenges

The greatest challenge we faced when working with the mouse connectome was understanding how to translate the available datasets into a single edge list to use as input for the various graph libraries in the code. While we were able to learn a lot about graph theory throughout this course, nobody in our group was particularly experienced with neuroscience, so we each had to read over the research paper multiple times to get even a basic understanding of what we would be working with. At first attempt, dataset S2 was used to construct the edge list. Each row was parsed for columns B and D to be written to the edge list; however, this resulted in a network with way more vertices than anything mentioned in the paper. It was then discovered that there was an auxiliary column which contained either a ‘yes’ or ‘no’ to indicate whether the reported connection was considered in the researcher’s final analysis. The parsing script was modified to only write values to the edge list if the connection was marked ‘yes’ for being considered, but there were still more connections than the paper described and visualizations for the network appeared to be fully connected. There was an obvious discrepancy between the handling of the dataset and the metrics reported in the paper, so we scanned thoroughly over the text to try and discover the key to the problem. The solution was in the caption of a figure in the paper - regions appeared on both sides of the mouse brain and the edgelist had to reflect this dichotomy. This was, after all, a connectome *between* gray matter regions in the mouse brain. The solution was to append the names of the side from columns A and C to each corresponding region in columns B and D to ensure they would appear unique in the resulting edgelist. The resulting edge list created a network with 308 vertices and 11,858 edges, which was mentioned in the paper as the total number of possible commissural connections for both domains of the mouse brain.

This edge list was used in the creation of the VTURCS poster and made for a pretty interesting 3D rendering of the mouse connectome. Since we were pressed for time to prepare for the symposium, it was great to finally have an edge list reflecting values mentioned in the paper; however, after speaking with our professor at the symposium we realized that we wrongly treated the network as an unweighted graph. The connectome is supposed to be weighted such that edge weights correspond to the strength of the connections between regions of the brain. There was not a clear indication of edge weights in dataset S2, but we found that dataset S3 contained multiple sheets with various adjacency matrices. The final edge list with weights was parsed from the sheet titled “CTX 6 module binned data” and resulted in a network with 154 vertices and 5394 edges - the exact numbers reported in the paper for all regions in both sides

of the brain. With this final dataset available, we could move on to proper analysis and visualization of the network.

We'd like to also mention that we did not have any background in neuroscience related concepts coming into this Capstone. Many of the graph theory concepts discussed in class were easy to follow along, but we lacked in the foundations of structure and function of the complex brain network. We saw relationships with graphs, but it was challenging to understand the functions of the different regions and how they corresponded with each other. One of the hardest parts was trying to understand the connection between the work being done and the depth of analysis that could be conducted on that work. For example, we had not realized that the maximal clique that was found also contained the most number of nodes that were overlapping between all communities. We were only able to find this relationship by sheer luck.

Another challenge we faced during this project was reading a large number of academic papers. Before this semester we rarely read research papers and when we did it was usually just the abstract or a summary. We had to read over the mouse connectome paper multiple times just to understand its contents in full, but found it time-consuming to learn about topics referenced from previous research. We feel as though this may be a common phenomenon in the research world, but whenever we didn't understand a term or algorithm mentioned in a paper, we would find the paper it was referenced from and read through that as well. This often took us down a 'rabbit hole' of academic research, and before we knew it we were reading deep into seemingly unrelated topics just to understand a small reference in the original paper.

Looking Back

It was interesting to see the different ways computer science can intersect with neuroscience. We were able to learn many different algorithms that provide deeper insight into advanced graph theory and correlate them with real mammal networks. We were also not experts in biology or neuroscience, but we used our knowledge of graph theory and programming to analyze the networks so that we could find what it means for different regions of the brain to overlap. Before this project began, we thought that we may be able to discover some interesting things about the mouse through graph theory, but after extensive analysis we discovered that many of these algorithms reveal network properties that are not one-to-one with the functional structure of a brain. Although the datasets we worked with were relatively small compared to other brain networks of larger mammals, the data itself was still very rich and a lot of analysis could be made on them. We would have definitely liked to have more algorithm's that we examined in terms of their complexity. We might have also had more time to look into rich clubs or other related concepts as well to tie into the mouse connectome. If we had more time, we would have also tried to find the maximal clique offset by 1, 2, 3, etc. to find the highest couple of k-cliques within the given data set.

What Each Member Did

Each member of our team worked on writing and editing this report. Contributions towards different aspects of the implementation are specified below.

Tom Evans: Processed mouse data from PNAS article - written to edgelists and region mappings, Wrote scripts for 3D visualizations of networks, generated figures for VTURCS poster and prepared symposium presentation, implemented COPRA algorithm for unweighted networks in Python.

Kavin Aravind & Rishi Pulluri: Worked extensively on analysis, running Louvains, Conga, CFinder, COPRA, and plotting degree centrality, frequency of each degree, max_clique, and 2d renders of networks from Figure 5A.

Sources

1. Swanson, Larry W., et al. “Organizing Principles for the Cerebral Cortex Network of Commissural and Association Connections.” PNAS, National Academy of Sciences, 2 Nov. 2017,
2. www.pnas.org/content/early/2017/11/01/1712928114.Sporns, Olaf & Zwi, Jonathan. (2004). The Small World of the Cerebral Cortex. Neuroinformatics. 2. 145-62. 10.1385/NI:2:2:145.
3. J.W. Scannell, G.A.P.C. Burns, C.C. Hilgetag, M.A. O'Neil, M.P. Young; The Connectional Organization of the Cortico-thalamic System of the Cat, Cerebral Cortex, Volume 9, Issue 3, 1 April 1999, Pages 277–299, https://doi.org/10.1093/cercor/9.3.277
4. Felleman DJ, Van Essen DC. Distributed hierarchical processing in the primate cerebral cortex. Cereb Cortex. 1991 Jan-Feb;1(1) 1-47. doi:10.1093/cercor/1.1.1. PMID: 1822724.
5. Steve Gregory (2010). Finding overlapping communities in networks by label propagation. New Journal of Physics, 12, 103018.
6. Vogt, B. A. (1976-09-01). "Retrosplenial cortex in the rhesus monkey: a cytoarchitectonic and Golgi study". *The Journal of Comparative Neurology*. **169** (1): 63–97.