Sarbjot Mann

FASS 210: Assignment 1
A Phonological Network

May 27th 2021

After loading the adjacency list *vitevitch.adjlist*. Using the networkx and matplotlib.pyplot python libraries the following graph as shown in Figure 1 was outputted.

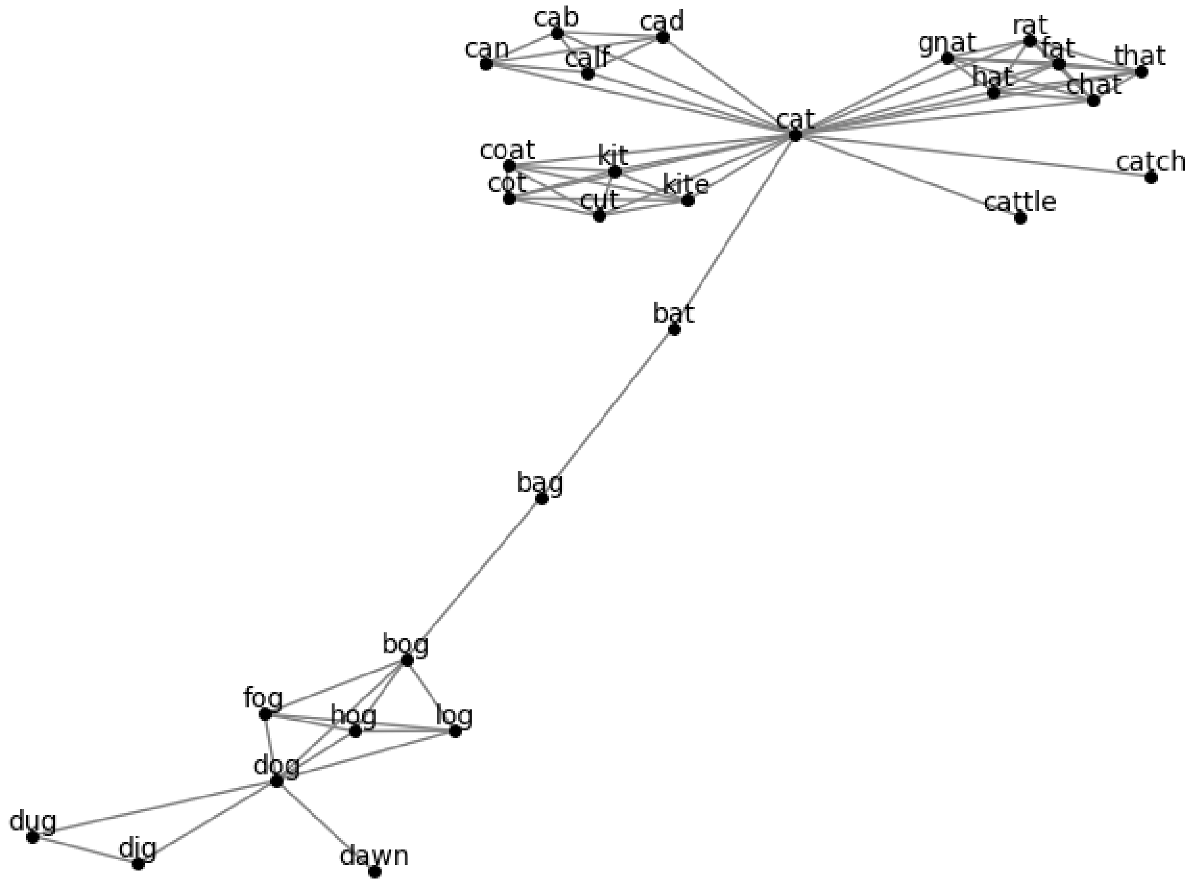

Figure 1: Phonological Network

After writing code which is available on Appendix 1 we were able to calculate the total number of nodes and edges, calculate the number of possible edges, density of the network, average degree of the network and find out what vertex have the highest and lowest degree.

The network consists of 65 edges and 28 nodes.The total possible edges in a undirected graph with 28 nodes is 378. The density and average density of our network is 0.17196 and 4.642857 respectively. The minimum degree of our network is 1 and the nodes labeled dawn, cattle, catch all fit this criteria. The maximum degree is 18 and the node cat fits this criteria.

The nodes themselves are connected based on their phonologically similarities. The edge represents one phoneme difference. Additionally from what we learned in lecture, words that have a high degree are often hard for people to recognize. If participants in a perceptual identification task or auditory lexical decision task they would respond to words with a lower degree more accurately, in the perceptual task and respond faster to words with a lower degree in a auditory decision task

Appendix 1

```python
import networkx as nx
import matplotlib.pyplot as plt  # Since we are not using a notebook we will import like this


def main():
    graph = nx.read_adjlist('../Datasets/vitevitch.adjlist')
    nx.draw(graph,
            with_labels=True,
            node_color='black',
            node_size=16,
            font_size=10,
            verticalalignment='bottom',
            edge_color='grey',
            )

    edges = number_of_edges(graph)
    nodes = number_of_nodes(graph)
    max_edges_possible = number_of_possible_edges(nodes)
    density = network_density(nodes, edges)
    max_degree = highest_degree(graph)
    min_degree = lowest_degree(graph)
    mean_degree = average_degree(nodes, edges)
    print(f"The number of edges in our graph is {edges}, and the number of nodes in our graph is {nodes}. \n"
          f"The max amount of edges possible in a undirected graph with {nodes} nodes is {max_edges_possible} \n"
          f"The density of our network is {density}. \n"
          f"The average density of our network is {mean_degree} \n"
          f"The nodes with the minimum degree are:")
    for nodes in min_degree:
        print(f"\t-{nodes} (degree of {graph.degree(nodes)})")
    print(f"The nodes with the maximum grades are: ")
    for nodes in max_degree:
        print(f"\t-{nodes} (degree of {graph.degree(nodes)})")
    plt.savefig('graph.png')
    plt.show()


def number_of_edges(G):
    return G.number_of_edges()


def number_of_nodes(G):
    return G.number_of_nodes()


def number_of_possible_edges(N):
    max_edges = (N * (N - 1)) / 2
    return max_edges


def network_density(N, L):
    numerator = 2 * L
    denominator = N * (N - 1)
    return numerator / denominator


def highest_degree(G):
    max_value = 0
    nodes_with_max_value = []
    for node in G.nodes():
        if G.degree(node) > max_value:
            max_value = G.degree(node)

    for node in G.nodes():
        if G.degree(node) == max_value:
            nodes_with_max_value.append(node)

    return nodes_with_max_value


def lowest_degree(G):
    min_value = 1
    nodes_with_min_value = []
    for node in G.nodes():
        if G.degree(node) < min_value:
            min_value = G.degree(node)

    for node in G.nodes():
        if G.degree(node) == min_value:
            nodes_with_min_value.append(node)

    return nodes_with_min_value


def average_degree(N, L):
    return (2 * L) / N

if __name__ == '__main__':
    main()
```