

Synaptic Pruning Mechanisms in Learning

Abstract

Synaptic pruning is the process of removing synapses in neural networks and has been considered to be a method of learning. During the developmental stages of the brain, synaptic pruning helps regulate efficiency and energy conservation. However, a destructive algorithm seems to be counter-intuitive to “learning.” When applied to association networks, pruning networks show evidence of learning associations when given good input. However, the pruning algorithm can be detrimental to network size and even limit learning for networks that are too small or have too high of a frequency threshold. For small networks, implementing constructive algorithm in addition to the pruning one may help ensure that a network continues to grow and learn.

Introduction

At the time of birth, the human brain consists of roughly 100 billion neurons in the cortex (The University of Maine, 2001). During the early stages of development (between infancy and adolescence), the weight and size of the brain increases significantly (by up to a factor of 5). This increase in density is not contributed to the creation of a large number of new neurons (neurogenesis), but rather an exponential increase in synaptic growth, known as *exuberant synaptogenesis* (Huttenlocher & Drabholkar, 1998). At infancy, each neuron averages around 2,500 synapses and at the peak of exuberant synaptogenesis (around 2-3 years of age) the number increases to around 15,000 (The University of Maine, 2001).

Shortly after this period of synaptic growth, the network experiences a decline in synaptic density as neuron connections that are used least-often are removed from the network.

The process of eliminating/weakening the irrelevant and inefficient synapses during this developmental period is known as *synaptic pruning*. Synaptic pruning has been viewed as a learning mechanism, where the “pruning” or removal of a synapse is dependent on the neurons’ responses to environmental factors and external stimuli (Craika & Bialystokb, 2006). The processes of synaptogenesis and synaptic pruning in the developmental stages of the brain have been associated with an individual’s “critical period,” a time where an individual is highly receptive to environmental stimuli and, therefore, learning (Drachman, 2005).

Many algorithms used to implement learning have utilized a bottom-up approach, where networks and memory usage increase with input over time. However, synaptic pruning utilizes a top-down, destructive algorithm to shrink the connections in a network, eliminating connections between neurons rather than creating them. The terms “destructive” and “learning” appear to be counter-intuitive, so this brings us to the question – what are the cost and benefits of a destructive algorithm, such as synaptic pruning, in a learning network?

One clear benefit is energy regulation and conservation. Consider that the brain of a three year old can contain approximately one quadrillion synapses (The University of Maine, 2001). The activity and maintenance of these neurons and their synapses is a wasteful use

of space and energy, of which the brain already consumes a significant amount. For reference, the human brain uses 20% of the body's energy, but makes up merely 2% of the body's total weight (University World News, 2013).

The focus of this paper is to describe and analyze the process and results of implementing the mechanisms of synaptic pruning to an association network. By analyzing the costs and benefits of this biology-inspired algorithm, we can observe its potential contributions to areas in machine learning. Using various text inputs, I was able to apply the synaptic pruning algorithm to a word association network and observe local and global changes in connections.

Creating a Network

In order to best observe the effects of the synaptic pruning algorithm, a highly connected network was needed to start with, comparable to a neural network after the exuberant synaptogenesis phase mentioned above. The goal was to create a network based off coherent text input, where word associations could be observed. Using a portion of the selected text input, exuberant synaptogenesis would be simulated by creating a large number of connections, or edges, between nodes, the details of which will be described later. In the context of this paper, a node refers to a word in a text and an edge represents an association.

In the case of neural networks, synapses would be considered a directed edge (the connection need not be reciprocal). However, in order to simplify the network and analysis, the network will assume that an edge, or association, is undirected. In other

words, an edge between node a and node b signifies that node a is associated with node b and node b is associated with node a . Having an undirected network is also ideal in the context that our network structure is based on (The University of Maine, 2001) word associations, in which variations in sentence and word arrangement would have a large impact if it were a directed network.

Initially, the network was designed so that every word in the network was connected to every other word in the network. However, this was obviously very inefficient in terms of memory usage and running time. Additionally, it was not comparable to a neural network after the exuberant synaptogenesis phase (the ratio between the total number of neurons and average number of synapses per neuron would have been too high compared to the neural network).

In another attempt to create a highly connected word association network, the criterion for edge connection was based on a word's proximity to other words for a set threshold. For example, for the input "The quick brown fox jumped over the lazy log" and for a threshold variable of 2, the word "fox" would be connected to the two words before it, "quick" and "brown," and the two words after it, "jumped" and "over." However, this method was discarded because it did not create much variability among the degree of nodes and was not effective around sentence structure and boundaries.

Finally, it was decided that a node's connections would be based off the sentence it was in. Using the example from above, the word "fox" would be connected to every other

word in the sentence. This definitely biases the degree of each node by the length of a sentence; however, it is more intuitive that the nodes corresponding to the same sentence are related to each other and creates a variation of degrees within the network.

A simple java program was created to “pretreat” the input text by removing grammatical characters (such as commas, apostrophes, etc.) and replacing characters indicating the end of a sentence (periods, question marks, etc.) with a new line character. This would allow for each sentence to be read in using the `readLine()` method and then split into words using the `split()` method and a delimiter. The program then outputted a file of .csv (comma-separated values) format, essentially defining each node and edge in the network. The .csv file could then be visualized and analyzed using the Gephi network visualizer software. The Gephi software arranges nodes and edges using a force-directed algorithm so that it can be best viewed in 2-D and 3-D formats. The effects of the software can be seen below, in which Figure 1 is the network before the force-directed algorithm is applied and Figure 2 is the network after.

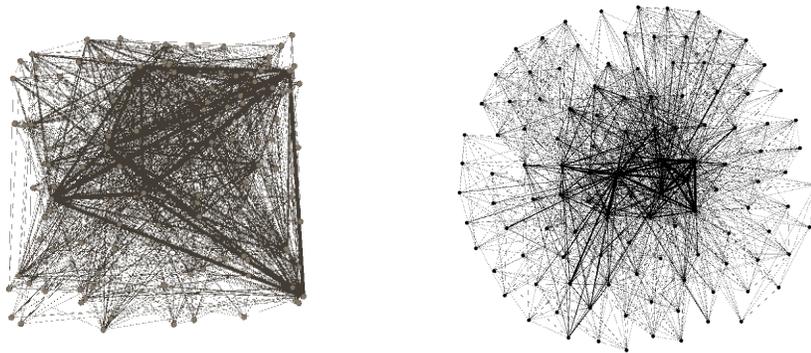


Figure 1 and 2: The results of the Gephi network visualizer software

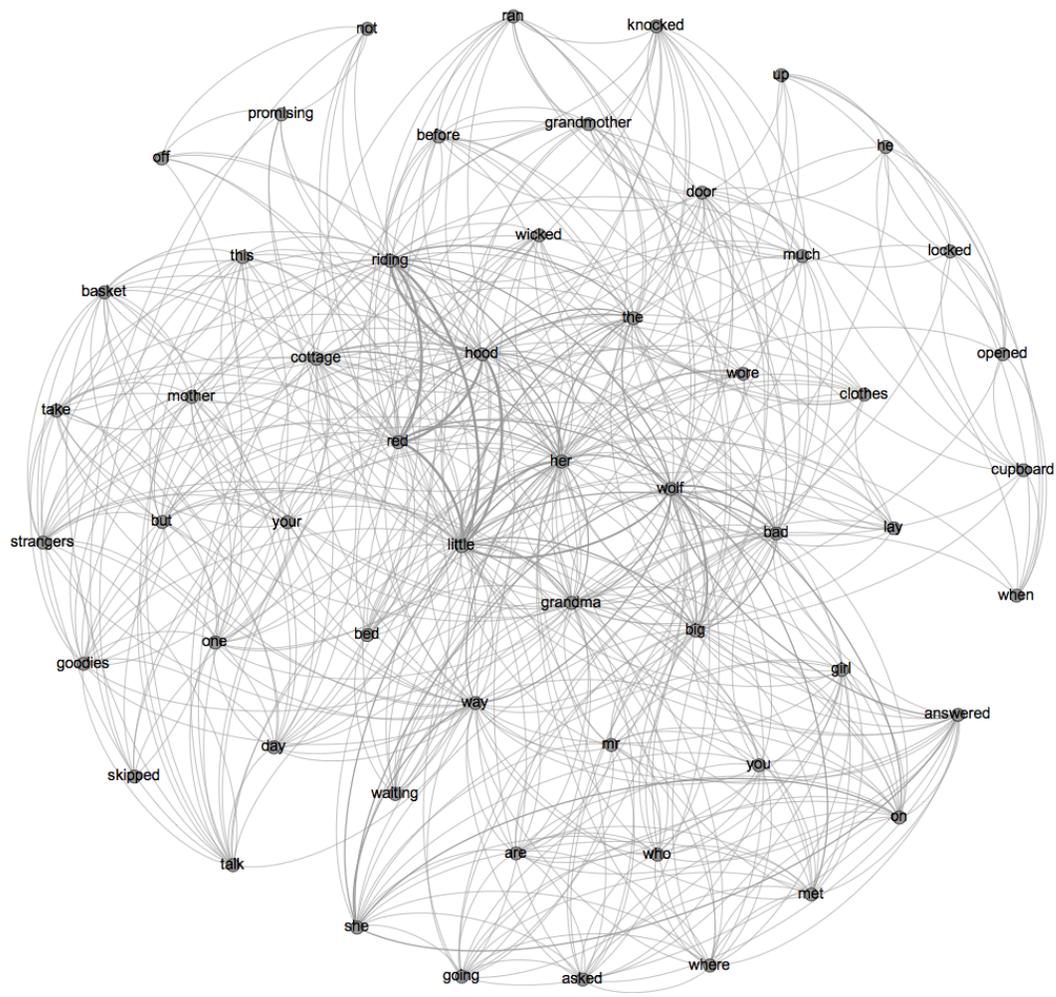


Figure 4. The network after removing commonly used words.

Note that the more often an edge occurs, the darker it appears. Compared to Figure 3, the numbers of nodes and edges have decreased due to the filtering of the input text. The simplification of the input was to further simplify the network.

Pruning the network

After the network is created, the second portion (in this case, the second paragraph) of the text is run through the same program, creating a list of associations for each

word according to the sentence that it is in, just as in the first half of the input. The initial network edges are then compared to the new list of edges generated from the second portion of the input. If any of the new edges match, it is added to the original edge list, increasing the frequency of an edge in the initial network (which in turn should darken the edge color). This output file is then run through another java program that consolidates the duplicates (with the edge node a to node b being equivalent to the edge from node b to node a) and then adds a frequency count for each edge. If the edge frequency is above the set threshold variable, then the edge is printed to a final text file (representing the final edge counts of our network). The final text file is then visualized through Gephi.

Figure 4 above describes the initial state of the network before running the synaptic pruning algorithm. It contains 64 nodes and 674 edges. It is a network trained using only the first paragraph of the Little Red Riding Hood text. The edges between the words “little,” “red,” “riding,” and “hood” are the strongest in the network, as would be expected. Figure 4 would also be the result of the pruning algorithm on the initial network if the threshold were set to 0.

Figure 5, and 6 below show the output network after various pruning simulation trails for threshold values of 2 and 3, respectively.

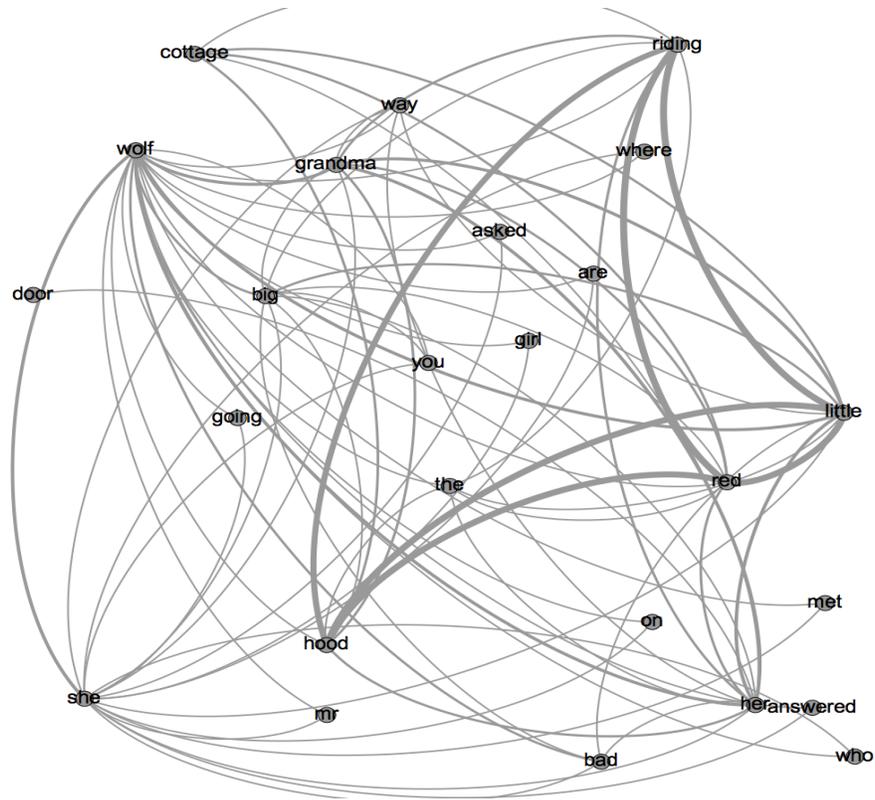


Figure 5. Network after Pruning Simulation for threshold = 2

The network in Figure 6 had 25 nodes and 78 edges. How did the number of nodes decrease if only connections are being eliminated? Consider the neural network. If a neuron had no synapses to other neurons, it would have no effect on the network and essentially be dead. In the case of Figure 5, the nodes that disappeared had all of its edge frequencies below the threshold. The threshold represents learning sensitivity. If there is a high threshold, more occurrences are required before synapses are removed for “learning” to occur. If the threshold is low, the occurrence of an edge has more value to the network, and learning is made easier.

Also to be observed in Figure 5, is the degree of the nodes relative to their edge frequency. The word “wolf” appears to have a high degree relative to the other nodes in the network. The fact that a node has a high degree is not very significant in the network. A node’s robustness to staying in the network is as much as its strongest connection. The edges between the words “little,” “red,” “riding,” and “hood” continue to have the strongest connections in the network. This is similar to the idea that “neurons that fire together, wire together,” also known as Hebbian learning. These words occur together in most instances and the network has learned their association with each other by increasing their weight (frequency count, in this case).

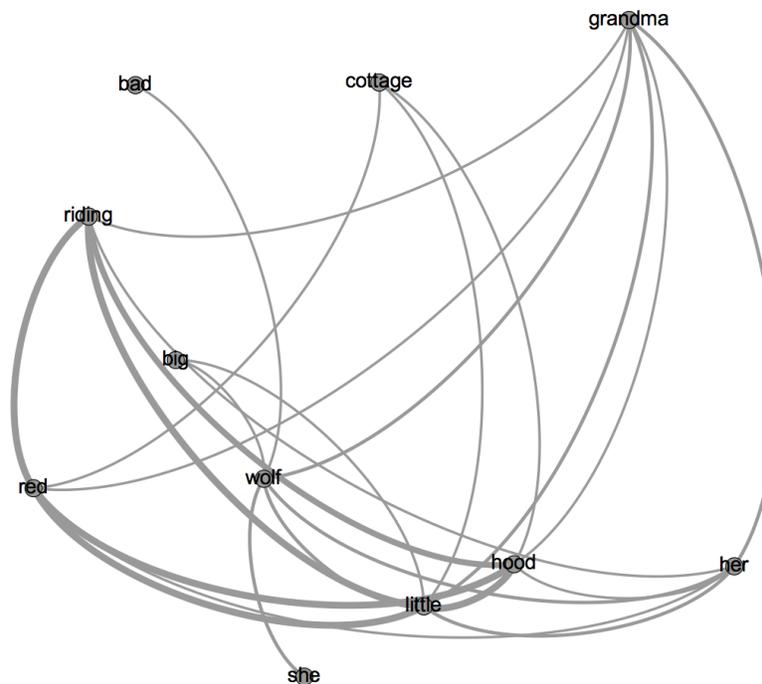


Figure 6. Network for the Little Red Riding Hood text with threshold = 5

The network in Figure 6 has 11 nodes and 25 edges. An increase of only one step in the threshold collapses the network to more than half its size in both edges and connections.

Analysis

The mechanisms behind synaptic pruning help promote efficiency by removing excess or unnecessary connections (and potentially nodes as well). This technique allows for robustness in nodes and connections that appear together, an effect of learning. If there are not enough occurrences of a pair or the occurrences are separated by too much time, learning becomes difficult. This algorithm appears useful for large amounts of nodes (i.e. neurons in the brain), in which destruction of connections or nodes has little impact on the entire network. For small networks, however, the process can be quite destructive and even limiting for low thresholds of removal. In these smaller network cases, such as the word association trial mentioned in this paper, it would be better for the algorithm to be paired with a constructive mechanism for creating new neurons and connections so that the network can have a potential for growth. The initial neural network in which the algorithm takes place is very important to the amount of information learned from the input. For example, in the case of the Little Red Riding Hood text, if there was a pair of words that was used in the second paragraph (representing external input) that was not used in the first paragraph (representing the initial network), the edge was simply ignored and the information was essentially lost. Therefore, the pruning algorithm on its own is very costly to small networks and further inhibits learning.

To further explore the effects of the algorithm, it may be interesting to experiment with time intervals in which the algorithm runs on a network. For example, the algorithm could run for every 100 words of input received. Similarly, integrating a constructive algorithm and attempting to find a balance between creating and removing nodes and synapses could be another point of exploration.

During the developmental stages, the brain experiences a large amount of growth. With a significantly large number of neurons and connections, it can afford to undergo excessive synaptic pruning for energy conservation purposes and the impact on the network is very slim. In areas such as machine learning, these types of memory capabilities have not yet been attained. This does not rule out its usefulness, however. Rather, it can be more useful since memory and energy is more valuable when it is limited.

Citations

Chechik, G., Meilijson, I., & Ruppin, E. (1997). Synaptic Pruning in Development: A Computational Account. MIT Press Journals .

Craika, F., & Bialystokb, E. (2006, March). Cognition through the lifespan: mechanisms of change. *Trends in Cognitive Sciences* .

Drachman, D. (2005). Do we have brain to spare? *Neurology* .

Huttenlocher, P., & Drabholkar, A. (1998). Regional differences in synaptogenesis in human cerebral cortex. *Journal of Comparative Neurology* .

The University of Maine. (2001). *Children and Brain Development: What We Know About How Children Learn*. Retrieved from Cooperative Extension Publications: <http://umaine.edu/publications/4356e/>

University World News. (2013). *The brain – Our most energy-consuming organ*. Retrieved from University World News: <http://www.universityworldnews.com/article.php?story=20130509171737492>

Ethics

I pledge my honor that this represents my work in accordance with University regulations.

Deborah Sandoval

Appendix:

Little Red Riding Hood Short Story Sample: <http://shortstoriesshort.com/story/little-red-riding-hood/>

One day, Little Red Riding Hood's mother said to her, "Take this basket of goodies to your grandma's cottage, but don't talk to strangers on the way!" Promising not to, Little Red Riding Hood skipped off. On her way she met the Big Bad Wolf who asked, "Where are you going, little girl?" "To my grandma's, Mr. Wolf!" she answered.

The Big Bad Wolf then ran to her grandmother's cottage much before Little Red Riding Hood, and knocked on the door. When Grandma opened the door, he locked her up in the cupboard. The wicked wolf then wore Grandma's clothes and lay on her bed, waiting for Little Red Riding Hood. When Little Red Riding Hood reached the cottage, she entered and went to Grandma's bedside. "My! What big eyes you have, Grandma!" she said in surprise. "All the better to see you with, my dear!" replied the wolf. "My! What big ears you have, Grandma!" said Little Red Riding Hood. "All the better to hear you with, my dear!" said the wolf. "What big teeth you have, Grandma!" said Little Red Riding Hood. "All the better to eat you with!" growled the wolf pouncing on her. Little Red Riding Hood screamed and the woodcutters in the forest came running to the cottage. They beat the Big Bad Wolf and rescued Grandma from the cupboard. Grandma hugged Little Red Riding Hood with joy. The Big Bad Wolf ran away never to be seen again. Little Red Riding Hood had learnt her lesson and never spoke to strangers ever again.

Gephi: The Open Graph Viz Platform: <http://gephi.github.io>