Kyle Janssen,

Aman Gill

CS 480

Spring 2014

**Homework #0: Part 2 - NETtalk**

Converting English text to speech has historically been a difficult task for computers. English pronunciation operates on a core set of rules, but also contains many context-based exceptions that would require extensive look-up tables to hard code. In the mid-80s Terrence J. Sejnowski and Charles R. Rosenberg developed a system to tackle this problem in a new way. They created a multilayered artificial neural network that attempted to learn the intricacies of the English language through practice.

The fundamental problem of converting text to speech is that of converting the basic units of text, letters and words, into the basic units of speech, called phonemes. Because phonemes correspond to a unique sound, and each letter of the alphabet does not, unique letters, or combinations of letters, can correspond to multiple phonemes depending on context. For a human, it is obvious that the letter “i” sounds different in the word “bit” than it does in the word “bite”, but to get an artificial system to make that distinction requires a bit more work.

* This next paragraph could benefit from an explanation of the formulas that determine the output of the processing units, and an explanation of the learning algorithm.

Sejnowski and Rosenberg’s approach was to use a three-layered hierarchical neural network, wherein each node, or processing unit, in a layer would give a single output based on its inputs. The base layer, or input layer, would read 7 characters at a time, send its output to the second layer, or hidden layer, which would in turn send its output to the final layer, the output layer, which produces the phoneme corresponding to the letter at the center of the 7 character input. The way the nodes would determine their output was by assigning the inputs from the previous level different weights, which began randomized. It was adjusting these weights up or down after each word was read, according to what gave the best results, which constituted learning for NETtalk. The use of the intermediate later of the neural network allowed for more adaptability by giving the network more of a faculty for decision-making.

The network trained on two data sets. One was a 1,024 word transcript of informal, continuous speech from a child, and the other was a set of 20,012 words from a dictionary. Both had phonetic transcriptions as well as English text. The weights in each node of the network began randomized, but were adjusted using a learning algorithm which would minimize the discrepancy between the ideal and actual output after each word was read. The idea was that as the network progressed through the data sets, it would continue to evolve and achieve an output closer to ideal. Performance of the network was measured by keeping track of the number of strictly defined “perfect matches”, which had to be within a small margin of the ideal output, and more loosely defined “best guesses”, which were simply the outputs that came closest to the ideal. Performance was also by feeding the output through a DECtalk system that would turn it into synthesized speech for human listening.

The sample of continuous, informal speech was difficult to learn from for obvious reasons. Similar words can be pronounced differently with minor variations in spelling and context. The data set was passed through the network repeatedly, with the first passes producing gibberish, but achieving better results with each pass. The output improved rapidly in the first 10,000 words, but continued to improve at a slower pace until after reading 50,000 words, the percentage of best guesses was 95% and the percentage of perfect matches was 55%. The trained network was then tested on a second sample of informal speech from the same speaker and achieved 78% best guesses and 35% perfect matches, indicating that a significant amount of the learned behavior could carry over to other data sets.

The 1000 most common English words were selected to train a fresh network to test on the dictionary set and to observe the learning capacity of the network as the number of processing units in the hidden layer was varied. The most common words in English are filled with grammatical and phonetical exceptions, so this part of the training also served to test the adaptability of the network. The training was performed using 0, 15, 30, 60, and 120 hidden units up to a word count of 30,000. With no hidden units, the network achieved 82% best guesses and about 22% perfect matches. The performance rose with each increase in the number of hidden units to 98% best guesses and about 51% perfect matches with 120 hidden units. The network with 120 hidden units was then tested on the 20,012-word dictionary. Starting untrained, it achieved 77% best guesses and 28% perfect matches on the first pass. After 5 passes, performance had improved to 90% best guesses and 48% perfect matches.

The training of NETtalk produced progressively better performance the more work it did, up to a saturation point. That saturation point continued to get closer to 100% accuracy, though never achieving it, the more complex the underlying structure of the network became. With both the informal speech training and the dictionary training, it was demonstrated that the learned behavior could be applied to data that had not been practiced on while retaining a significant level of performance. It is mentioned that while learning, the network was processing 2 letters per second with their setup. With this level of performance being attained in the mid-80s, it is interesting to think about what kind of performance could be achieved with today’s hardware.

References:

1. Terrence J. Sejnowski and Charles R. Rosenberg, “NETtalk: a parallel network that learns to read aloud,” *The Johns Hopkins University Electrical Engineering and Computer Science Technical Report*, JHU/EECS-86/01, 32 pp.