Rebel Speech Recognition

Design Document (under construction)

Introduction

Engine Design

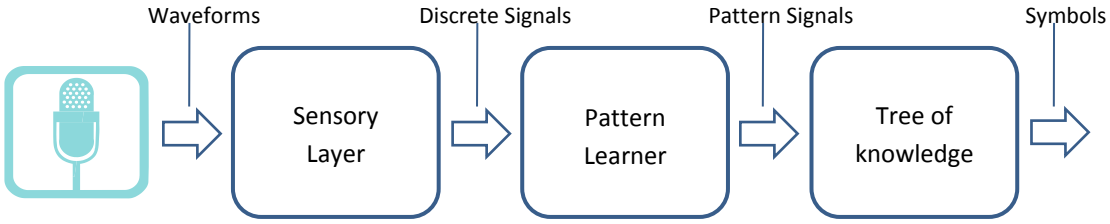
The Rebel Speech Recognition Engine is a biologically plausible spiking neural network designed for general, unsupervised audio learning and recognition. As such, it can learn to recognize speech in any language and to accurately recognize any type of sound, not just speech. It is designed to be noise and speaker tolerant while being faster (nearly instantaneous) and more accurate than existing technologies.

Features

One of the main characteristics that distinguish this engine from other speech recognizers is that, given adequate training, it can accurately recognize a sentence or a word before the human speaker has finished speaking it or even if parts of the sound are missing or garbled. This is not unlike the way we do it. Thanks to a novel, hierarchical memory architecture, it can also learn to recognize and latch on to a single voice in a roomful of voices. This is known as the cocktail party problem, one of the hardest problems in computer science. Another distinguishing feature is that, although the engine processes probabilistic sensory inputs, it is not based on Bayesian statistics (e.g., the hidden Markov model) but on a simple winner-take-all mechanism that uses little or no math.

Program Design

The engine consists of three software modules as depicted below.



The sensory layer uses a Fast Fourier Transform algorithm to convert audio waveform data into discrete signals. These raw sensory signals are fed directly into a pattern learner that combines them into multiple concurrent and unique groups called patterns. Pattern signals are then piped into a sequence learner/recognizer called the tree of knowledge (TOK) where they are organized into temporal hierarchies called branches. Each branch represents a specific sound or a sequence of sounds.

©2012 Louis Savain

Theory

Bayesian Bandwagon

The most surprising thing about the Rebel speech recognition engine is that, unlike current state of the art speech recognizers, it does not use Bayesian statistics. This will come as a surprise to AI experts because they have all jumped on the Bayesian bandwagon many years ago. Even those who claim to be closely emulating biological systems believe in the myth of the Bayesian brain. Of course, this is pure speculation and wishful thinking because there is no biological evidence for it. In a way, this is not unlike the way the AI community jumped on the symbol manipulation bandwagon back in the 1950s, only to be proven wrong more than half a century later. I have excellent reasons to believe that, in spite of its current utility, this is yet another red herring on the road to true AI.

Traditional Speech Recognition

Most speech recognition systems use a Bayesian probabilistic model, such as the hidden Markov model, to determine which senone, phoneme or word is most likely to come next in a given speech segment. A learning algorithm is normally used to compile a large database of such probabilities. During recognition, hypotheses generated for a given sound are tested against these precompiled expectations and the one with the highest probability is selected as the winner.

Rebel Speech Recognition

In contrast to the above, the Rebel Speech engine does not rely on pre-learned probabilities. Rather, it uses an approach that is as counter-intuitive as it is powerful. In this approach, the probability that the interpretation of a sound is correct is not known in advance but is computed on the fly. The way it works is that the engine creates a hierarchical database of as many sequences of learned sounds as possible, starting with tiny snippets of sound that are shorter than a senone. When sounds are detected, they attempt to activate various sequences and the sequence with the highest hit count is the winner. A winner is usually found before the speaker has finished speaking. It works because sound patterns are so unique, they form very few sequences. Once a winner is determined, all other sequences that do not belong to the same branch in the hierarchy are immediately suppressed. This approach leads to very high recognition accuracy even when parts of the speech are missing; and it makes it possible to solve the cocktail party problem (pdf).

©2012 Louis Savain

Sensory Layer

Discrete Signals and Population Coding

The sensory layer does the thankless but vital job of converting audio phenomena into the kind of sensory signals that the engine can process. It uses a simple FFT algorithm to convert batches of digital audio waveform data into discrete signals representing changes in various frequencies and amplitudes. The design of the sensory layer is based on the notion that what is important to an intelligent system is not so much the state of the word but how the world changes. The best way to accomplish this is to generate a constant stream of data by rapidly transforming detected changes into discrete signals.

Rebel Speech uses a discretization method called *population coding* to encode amplitude. Essentially, a fixed number of discrete neurons are assigned to each frequency and the number of neurons that fire at any given time is proportional to the amplitude and speed of the change. The use of population coding is absolutely crucial to the proper operation of the pattern learner and the tree of knowledge because the fine temporal structure of the changes, a must for pattern and sequence learning, is preserved during the transformation. One of the advantages of this method is that the coarseness of the conversion can be adjusted so as to optimize response speed and recognition accuracy for a given computer.

Some Early Specifications

Currently, the microphone sampling rate is set at 11 KHz and each sampled batch contains 1024 bytes of audio data. After conversion, the data is transformed into 512 values, each representing the amplitude of a unique frequency. As of now, the sensory layer uses only the lowest 24 frequencies. It turned out that 24 are enough for most speech recognition tasks. Using more would only slow down the engine without significantly adding to its recognition accuracy. These specifications are subject to change.

To be continued…

©2012 Louis Savain

Pattern Learning and Recognition

Patterns

A pattern is a group of concurrent sensory signals. It represents a unique recurring phenomenon in the environment. By definition, every signal in a pattern must have a different sensory origin, i.e., they must arrive in different parallel sensory streams.

> *Note: The neuroscience literature uses the term 'spatial' to refer to a concurrent pattern and 'temporal' to refer to a sequence of patterns. I am opposed to the usage because it is misleading to the uninitiated. There is nothing spatial about patterns. From the point of view of the intelligent system, a pattern is just as much a temporal entity as a sequence. A sound, for example, consists of multiple sine waves and these have nothing to do with spatiality. Yet the brain uses identical learning principles for both visual and auditory stimuli.*

Pattern Learner

Pattern learning is based on the assumption that there are temporal regularities in the environment that are reflected in the sensory signals. The pattern learner contains a large number of pattern recognition neurons that receive signals from the sensory layer. It tries to discover as many patterns in the sensory space as possible. It accomplishes this task by arranging patterns to form a hierarchy.

The pattern hierarchy sends its output signals directly to the sequence learner. Signal propagation within the pattern hierarchy must be fast enough to occur within a single cycle or about 10 milliseconds. From the point of view of the rest of the system, signal propagation within the hierarchy is instantaneous.

The pattern learner uses an unsupervised learning mechanism that is based on a few simple but strict rules[1]. These rules are essential to the proper functioning of the engine as a whole.

Pattern Recognition

Pattern recognition neurons fire when they detect a specific event. In a perfect sensory situation, a pattern neuron would fire if and only if all of its inputs fired simultaneously. Unfortunately, this rarely happens due to the uncertainties of sensory phenomena. So a pattern neuron will fire every time a certain number of its inputs fire concurrently. This number is called the firing threshold and can be adjusted to suit the environment and the requirements of the system's designers. The strength of the emitted signal is proportional to the number of inputs that fired. It is a measure of how certain the pattern recognition neuron is that the event actually occurred.

To be continued…

[1] Unfortunately, I cannot reveal the rules at this time because they are trade secrets that I plan to use to raise money for my AI research. Sorry.

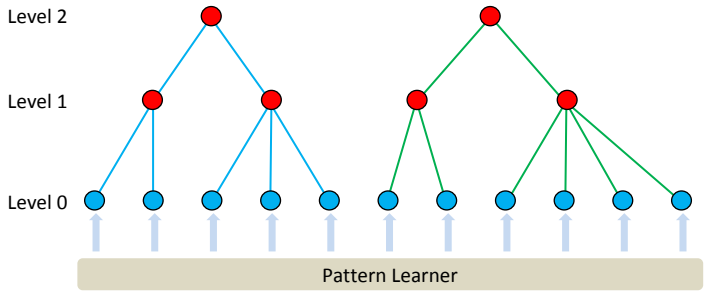
©2012 Louis Savain

©2012 Louis Savain

The Tree of Knowledge

Branches and Leaves

The TOK is a self-assembling, hierarchical memory structure that receives its inputs from the pattern module. The main function of the TOK is to learn to recognize sequences. Metaphorically speaking, patterns are the leaves of the TOK. A branch is either a sequence of patterns, a sequence of other sequences or a combination thereof. A branch can have up to seven sequential nodes. It is the fundamental building block of the TOK.



Hierarchical Memory

In the illustration above, patterns are shown as blue circles; the red circles represent branches and sub-branches. Branches compete for activation using a winner-take-all mechanism. That is to say, the branch that receives the strongest signals wins and the others are suppressed. Audio recognition in the TOK consists of activating one branch (and its sub-branches, if any) at a time. Recognition is highly invariant to changes in speakers, accents, volume, pitch, noise level and timing.

Sequence Learning

Sequence learning in the TOK is as simple as it is powerful. This is because most of the learning has already been done by the pattern learner. As mentioned previously, patterns are so unique that they can form only a very limited number of sequences. In fact, most of the time, a pattern will permit a single successor and/or predecessor. In this light, sequence learning is mostly a recording process since any sequence is a good sequence.

To be continued…

Causal Reasoning

Although causal reasoning is not a part of this engine, I think it is important that I mention it here because the TOK is the main focus of my on-going work on universal artificial intelligence. There is no getting around the fact that sensory perception is necessarily probabilistic. However, causal reasoning within the TOK is purely deterministic. In other words, a branch is either on or off; it is never partially

©2012 Louis Savain

activated. So-called Bayesian reasoning is a red herring in my opinion. The deterministic nature of reasoning is something that other AI researchers, especially Judea Pearl, have had to come to terms with in spite of the current craze in favor of purely probabilistic processes. See this recent Cambridge University Press interview with Pearl for a glimpse at his work on causal thinking.

Memory and Cortical Columns

The TOK is not just a sequence recognizer. It encapsulates both short and long term memory, which includes episodic memory. That is to say, it records not just the order of events but also their precise temporal intervals. Without this capability, accurate prediction would be impossible. It goes without saying that there is more to a sequence than just temporal order.

To be continued…

©2012 Louis Savain