Sparse Distributed Memory

A study of psychologically driven storage

Pentti Kanerva

Outline

- Neurons as address decoders
- Best match
- Sparse Memory
- Distributed Storage

Goal

The goal of Kanerva’s paper is to present a method of storage that, given a test vector, can retrieve the best match to the vector among a set of previously stored vectors. He builds this model from the ground up, first using neurons as address decoders. He then describes a method of storage called sparse memory. This fails his goals, however, he builds on this model, culminating in what he calls distributed storage.

Goal

Kanerva believes that his model may be a good generalization of how the human brain works. It is non-conventional in that it does not strive to provide a method for storing exactly that which one would like to retrieve. Rather, it provides a way to retrieve those vectors that are very similar to a given vector.

Address Decoder Neurons

- Have standard weight vector [called input coefficients]
  - Bipolar
- Linear Threshold Gate
- Unipolar Output

Key Observation

A one-dimensional weight vector can be realized as an address, for which the neuron can ‘code’ for.

The threshold value can be realized as the difference in the desired address and the input address.

The *Response Region* of this type of neuron is the set of inputs for which the output of the neuron is 1.

Mathematical Aside

It isn't necessary that the weights be unipolar, but it makes the model simple, and more useful. Also, it makes much more sense that each bit of the address contributes equally, otherwise the distance function becomes skewed.

Biological Aside

The addition and deletion of new synapses will not affect greatly the address of an individual neuron. [This corresponds to the expansion or contraction of the weight vector.] Furthermore, if synapses changed from excitatory to inhibitory or vice versa, the address would be affected, but neurons are not known to make this sort of change.

Assumption for Rest of Work

- Weights are constant over time
  - Contributing to the idea that a neuron has an address
- Thresholds are allowed to vary

Best Match

Problem: Given a data set of stored words, retrieve the best match to a test word.

Formally: Describe a filing scheme for storing a set X of words so that one can retrieve, in minimum time, the stored word ζ that is the best match [closest in Hamming Distance] to a test word z.

Recall: Hamming Distance between two binary sequences is the number of bits that disagree.

Problem with Conventional Architecture

It is quicker to check each word of a trillion-word data set against each word of a trillion-word test set, than it is to load into memory the given data set.

Kanerva’s ‘Machine’

An array of address decoding neurons, with the ability to change all their thresholds as a unit. Output probably much like a hardware architecture, where the closest match will come out, when the threshold is passed.

Sparse [Random Access] Memory

- Storage locations and addresses are given from the start.
- Only contents of locations are modifiable
- Storage locations are very few in comparison with $N = 2^n$
- Storage locations randomly distributed over $\{0, 1\}^n$

Definitions

Hard Locations: The random sample of actual storage locations.

Nearest N'-Neighbor, x': The hard location closest to $x \in N$.

An example of a sparse memory

- Address space $N = \{0,1\}^{1000}$
- Hard Locations $N' = 1,000,000$
- On average, each location $x \in N$ is 424 bits from x'

Best Match w/ Sparse Memory

Let X be a random data set of 10,000 words.

Try storing each $\xi \in X$ in ξ'

With test word z and ξ closest to z in X, what is the probability that the occupied hard location nearest to z contains ξ?

Note: we are searching for ξ, located in ξ', and we are not looking for z' since z is not necessarily in X.

Distributed Storage

- Many storage locations participate in a write/read operation
- Appearance of a RAM with large address space
- If η is stored at ξ, then reading from ξ retrieves η
- Claim: Reading from address x similar to ξ retrieves a word y that is closer to η than x to ξ.

Definitions

Access Radius, Circle: If r is the **access radius** then given address x, we access all hard locations within r of x when reading from or writing to x.

Contents of a Location x: Each storage location will contain all of the words that have ever been written there.

Data of x, D(x): The data retrievable from address x is the multiset of all the words in the hard locations inside the access radius of x.

Reading at x: Returns a single word.

Best Match

Given a test word z retrieve target word $\zeta \in X$

1. Search $D(z)$ for best match
2. Most frequent word of $D(z)$
3. Random word of $D(z)$
4. Compute archetypical element of $D(z)$

Finding the Best Match

Key Observation: If ζ is the word of X most similar to the test word, we can expect it to occur the most often in $D(z)$.

Obvious Attempt: Take the word that occurs most frequently.....Too much computation.

Weak Attempt: Take a random word.....Doesn't work.

Working Attempt: Take the average element....Works....In most cases.

Computing the Archetypical Element

The Average Element: Take the bitwise sum of $D(z)$.

If $d(z, \zeta) < 209$ than in a series of steps the bitwise sum of each successive word read will converge on ζ.

Other Properties

- Convergence/Divergence
- Memory Capacity
- Storage Location Capacity
- Signal Strength/Noise/Fidelity

What's it got to do with Neural Networks?

Knowing that one knows may be indicated by fast convergence.

Tip-of-the-tongue may be indicated by the critical point, at which we may recover the memory, and we may not.

Rehearsal of some action, might be indicated by the storage of that item over and over.

The Effects of brain damage which sometimes seem non-existent, may be indicative of a distributed storage, where it is still possible to retrieve memories.

Questions

???