Clustering with Monadic Memory

Peter Overmann
15 Aug 2022

This notebook demonstrates how to use Monadic Memory for clustering/pooling a large number of SDRs.

Monadic Memory Algorithm

```
MonadicMemory[f_Symbol, {n_Integer, p_Integer}] :=
 Module[ {overlap, D1, D2, items = 0},

  DyadicMemory[ D1, {n, p}];
  DyadicMemory[ D2, {n, p}];

  overlap[a_SparseArray, b_SparseArray] := Total[BitAnd[a, b]];

  (* random SDR *)
  f[] := SparseArray[ RandomSample[ Range[n], p] → Table[1, p], {n}];

  (* store and recall x *)
  f[x_SparseArray] := Module[ {r, hidden},

    r  = D2[D1[D2[D1[x]]]];

    If[HammingDistance[x, r] < p, Return[r]];

    items++;
    hidden = f[];
    D1[ x → hidden]; D2[ hidden → x];

    x
   ];

  f["Items"] := items;
 ]
```

Noise

Visualization

Configuration

```
Get[ $UserBaseDirectory <> "/TriadicMemory/dyadicmemoryC.m"]
```

```
n = 1000;
p = 20;

MonadicMemory[ M, {n, p}];
```

Generate Test Data

Generate k=100 random SDRs (“classes”)

```
k = 100;

classes = Table[ M[], k];
```

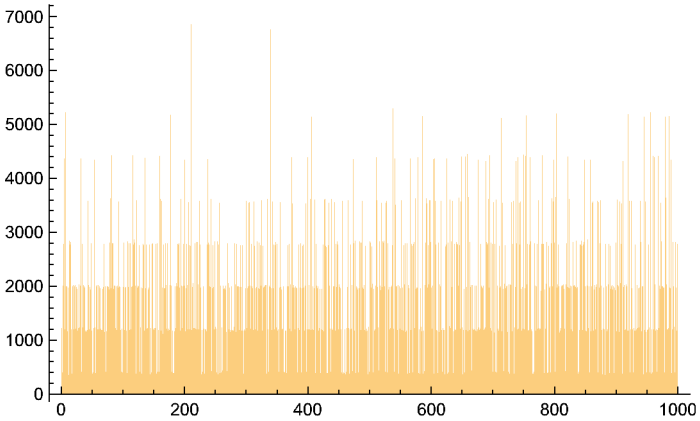
For each class, make 1000 variations with 4 random bits changed

```
data = RandomSample[
   Flatten[ Table[ SDRNoise[ SDRNoise[#, -4], 4], 1000] & /@ classes, 1]];

pos[x_SparseArray] := Sort[Flatten[x["NonzeroPositions"]]];

Export["data.tsv", pos /@ data];
```

Visualize the distribution of SDR bits in the dataset:



Write data set to a Monadic Memory

```
M /@ data; // AbsoluteTiming
```

```
{155.345, Null}
```

Number of hidden vectors created in the Monadic Memory (slightly more than the number of classes)

```
M["Items"]
```

```
105
```

Write dataset again

```
out = M /@ data; // AbsoluteTiming
```

```
{144.324, Null}
```

The number of stored items has slightly increased (the algorithm keeps learning during recall)

```
M["Items"]
```

```
106
```

Number of recalled items (same as length of dataset)

```
Length[out]
```

```
100 000
```

Number of **different** items in output -- this is the number of clusters found. Each SDR from the original dataset has been mapped to a representative SDR from one of the clusters found.

```
Union[pos /@ out] // Length
```

```
111
```