Union Pooler Psuedocode

BoltzmannBrain edited this page on May 11 · [7 revisions](https://github.com/numenta/nupic.research/wiki/Union-Pooler-Psuedocode/_history)

**Pages 22**



* [**Home**](https://github.com/numenta/nupic.research/wiki)
* [**AI & ML Reference Material**](https://github.com/numenta/nupic.research/wiki/AI-&-ML-Reference-Material)
* [**AI & ML Talks**](https://github.com/numenta/nupic.research/wiki/AI-&-ML-Talks)
* [**Behavior and Pooling Requirements**](https://github.com/numenta/nupic.research/wiki/Behavior-and-Pooling-Requirements)
* [**Capacity experiment: setup and validation**](https://github.com/numenta/nupic.research/wiki/Capacity-experiment:-setup-and-validation)
* [**Capacity Test Results**](https://github.com/numenta/nupic.research/wiki/Capacity-Test-Results)
* [**Capacity Test Results (obsolete)**](https://github.com/numenta/nupic.research/wiki/Capacity-Test-Results-(obsolete))
* [**Current Research Tasks**](https://github.com/numenta/nupic.research/wiki/Current-Research-Tasks)
* [**Extraneous Predictions**](https://github.com/numenta/nupic.research/wiki/Extraneous-Predictions)
* [**Installation on EC2**](https://github.com/numenta/nupic.research/wiki/Installation-on-EC2)
* [**Meeting and Progress Notes**](https://github.com/numenta/nupic.research/wiki/Meeting-and-Progress-Notes)
* [**Meeting Notes from September 22**](https://github.com/numenta/nupic.research/wiki/Meeting-Notes-from-September-22)
* [**Modification to sensory motor temporal memory (Layer 4) algorithm**](https://github.com/numenta/nupic.research/wiki/Modification-to-sensory-motor-temporal-memory-(Layer-4)-algorithm)
* [**Open Questions**](https://github.com/numenta/nupic.research/wiki/Open-Questions)
* [**Overview of the Temporal Pooler**](https://github.com/numenta/nupic.research/wiki/Overview-of-the-Temporal-Pooler)
* [**Research Glossary**](https://github.com/numenta/nupic.research/wiki/Research-Glossary)
* [**Sensorimotor Inference Algorithm**](https://github.com/numenta/nupic.research/wiki/Sensorimotor-Inference-Algorithm)
* [**Sensorimotor Inference Algorithm Discussion**](https://github.com/numenta/nupic.research/wiki/Sensorimotor-Inference-Algorithm-Discussion)
* [**Temporal Pooler Overview**](https://github.com/numenta/nupic.research/wiki/Temporal-Pooler-Overview)
* [**Temporal Pooler Simultaneous Tandem Learning Experiments**](https://github.com/numenta/nupic.research/wiki/Temporal-Pooler-Simultaneous-Tandem-Learning-Experiments)
* [**Union Pooler Experiment Results**](https://github.com/numenta/nupic.research/wiki/Union-Pooler-Experiment-Results)
* [**Union Pooler Psuedocode**](https://github.com/numenta/nupic.research/wiki/Union-Pooler-Psuedocode)
* [Show 7 more pages…](https://github.com/numenta/nupic.research/wiki/Union-Pooler-Psuedocode)

**Clone this wiki locally**



[**Clone in Desktop**](https://windows.github.com/)

**Union Pooler Algorithm Pseudocode**

Note: This algorithm is a work in progress. (5.11.15)

**Data Structures**

activeInput ­- Binary vector representing active cells from Temporal Memory

predictedActiveInput - Binary vector representing correctly predicted active cells from Temporal Memory

poolingActivation - A vector of scalar activation values for each cell in the Union Pooler

**Parameters**

activeOverlapWeight - ­Weight given to overlap due to activeInput

predictedActiveOverlapWeight ­- Weight given to overlap due to predictedActiveInput

activationFunction -­ A function that updates a pooling activation value based on an excitation or a decay. Possible forms include linear, exponential, or logistic.

maxUnionActivity -­ Maximum allowed size of the union SDR as a percentage of the number of Union Pooler cells

**Algorithm**

# Compute current active cells based on current input

activeOverlaps = calcOverlap(activeInput)

predictedActiveOverlaps = calcOverlap(predictedActiveInput)

totalOverlap = (activeOverlaps \* activeOverlapWeight + predictedActiveOverlaps \*

predictedActiveOverlapWeight)

boostedOverlaps = boost \* totalOverlap if learn else totalOverlap

activeCells = inhibitColumns(boostedOverlaps)

# Perform Spatial Pooler learning algorithm with boosting

adaptSynapses(activeInput, activeCells)

updateDutyCycles(totalOverlap, activeCells)

bumpUpWeakColumns()

updateBoostFactors()

if isUpdateRound():

updateInhibitionRadius()

updateMinDutyCycles()

# Update Union SDR based on new set of active cells; decrement pooling activation of all cells

decayFunction.decay(poolingActivation)

# Add to the poolingActivation of those active Union Pooler cells receiving active inputs

addToPoolingActivation(activeCells, activeOverlaps)

# Same for Union Pooler cells receiving active-predicted inputs

addToPoolingActivation(activeCells, predictedActiveOverlaps)

# Compute the current most salient cells in terms of poolingActivation.

# Cells with zero poolingActivation cannot win.

unionSDR = getMostActiveCells(poolingActivation, columnCount \* maxUnionActivity)

return unionSDR