# Analyze and document boosting algorithm

Information Technology Course

Module Software Engineering

by Damir Dobric / Andreas Pech

Your Name

youremail@stud.fra-uas.de

**Abstract— “Boosting” is a term which refers to a branch of algorithms which changes weak learners to strong learners. Boosting is a group strategy for improving the model forecasts of some random learning algorithm. Boosting is to prepare weak learner consecutively, each attempting to address its predecessor. This work shows the boosting algorithm which boosts the columns in spatial pooler (SP).** **The boosting algorithm tracks the column activity in spatial pooler(SP) and it makes it sure that all columns are uniformly used across all seen patterns** .**This paper shows several experiments like Adapt synapses, Update duty cycle, Bump up weak columns, Update boost factors, inhibition radius and min duty cycle of boosting algorithm. Each experiment is evaluated based on the theoretical and practical framework and then summarized the results in graphical diagrams. Abstract must sell your paper. People read abstract and decide if it is interesting or not. You have to spend a lot of time to write the abstract. It should describe the topic, the motivation and the cool result. It should be no longer than this example.**

***Keywords— Boosting Algorithm, Synapses,***

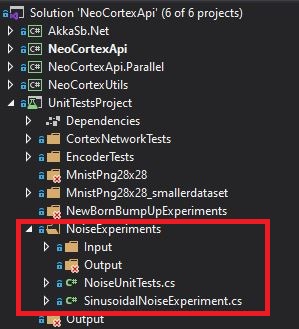
***Boost Factor, Duty Cycle, Your Keywords here.***

1. **INTRODUCTION**

The introduction explains why this research is important or necessary or important. Begin by describing the problem or situation that motivates the research. Move to discussing the current state of research in the field; then reveal a “gap” or problem in the field. Finally, explain how the present research is a solution to that problem or gap. If the study has hypotheses, they are presented at the end of the introduction.

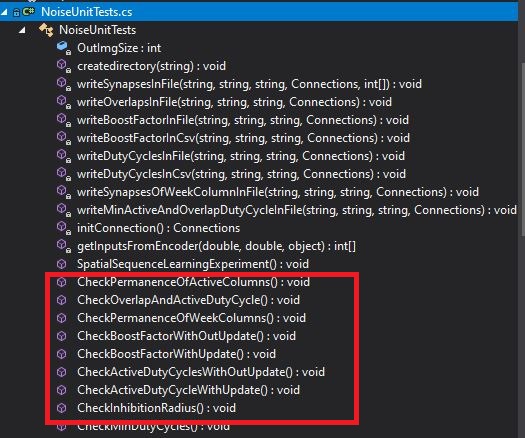
1. **METHODS**

The methods section tells readers how you conducted your study. It includes information about your population, sample, methods, and equipment. The “gold standard” of the methods section is that it should enable readers to duplicate your study. Methods sections typically use subheadings; they are written in past tense, and they use a lot of passive voice. This is typically the least read section of an IMRaD report.If you need to reference something use Add Citations from the menu [1] [2]



*Figure 1 This is the picture description.*

Under NoiseUnitTests, the unit tests performed are highlighted.



The results of the unit tests conducted in below folder:

se-cloud-2020-2021\MyProject\NeoCortexApi\ neocortexapi\NeoCortexApi\UnitTestsProject\bin\Debug\net5.0\boosting.

The results of the unit tests conducted in folder [3].

Initially we set these values for unit test:

|  |  |
| --- | --- |
| POTENTIAL\_RADIUS | 64\*64 |
| POTENTIAL\_PCT | 1 |
| GLOBAL\_INHIBITION | FALSE |
| INHIBITION\_RADIUS | 0.25 \* 64 \* 64 |
| NUM ACTIVE COLUMNS PER INH AREA | 0.1 \* 64 \* 64 |
| DUTY\_CYCLE\_PERIOD | 1000000 |
| MAX\_BOOST | 5 |
| Input Dimensions | 32, 32 |
| Column Dimensions | 64, 64 |

1. **RESULTS**

In this section, you present your findings. Typically, the Results section contains only the findings, not any explanation of or commentary on the findings (see below). Results sections are usually written in the past tense. Make sure all tables and figures are labeled and numbered separately.

Diagram

Description automatically generated

Fig. 1: Signal propagating down an axon to the dendrites of the next cells [1].

The concept of permanence is used in HTM to change the connectedness of synapses. Permanence is scalar value, and its values range from zero to one. This represents the degree of connectedness between the axon and the dendrite. The permanence value of zero signifies that this is not valid for potential synapse, and it cannot be a valid synapse. Typically, there is a threshold permanence value 0.2 which represents that a synapse is just connected but it is possible to get unconnected easily. If the permanence value is high for example 0.9 means that the synapse is connected, and it cannot be easily disconnected [3]. When the synapse’s permanence value is more than the threshold value, that synapse relates to weight one. If this value is below the threshold, that is unconnected with weight zero. In the classical neural networks, there is no individual weighting for synaptic connection. However, HTM has this feature to create and remove these connections. According to Hawkins, HTM achieves a higher data storage capacity by creating and extracting synaptic connections without changing the weights of permanent connections [1].

Diagram

Description automatically generated

Fig. 2 [1]

**Input 1 step 0:** Every input from encoder is internally run three times inside the unit test and is called step 0,1,2. We just took 1 input of many inputs to represent our results.

Columns whose value is 0.75 has been active for 75% of time. Columns whose value is 0.25 has been active for 25% of the time.

Fig 4. Input 1 step 0

**Input 1 Step 2:**

At input 1 and step 2 the value of columns are much improved than last time. Now more and more columns are getting active. The value of columns are increasing and so every column is getting active for a certain period of time as we can see it clearly on the chart below.

Fig 5. input 1 step 2

**3.Bump Up Weak Column:**

In the HTM determine the active columns of cells. Via the synapses on a proximal dendrite, each column is connected to a subset of the input bits. Subsets for different columns may overlap but they are not equal. For the activation of the columns different input patterns result in different levels. The columns with the strongest activation resist columns with weaker activation. A column’s inhibition area is adjustable, and it can be expanding from very small to the entire region. HTM learns by forming and unforming connections between cells. The permanence value of the active column’s synapses increases when it is connected to active bits and decrease otherwise. The higher permanence value represents the more overlap between the pattern represented by the columns and the input[1].

If any columns do not become active for a long period, it cannot learn anything. The columns whose activity level is too low, those columns permanence value can be increases via “BumpUpWeakColumns” methods. With the overlap duty cycle technique, it is possible to identify those columns, and it reinforces (increments) all the proximal synapses of columns that have lower average input compare to the others.

In the code for unit test, we did not get any value after doing BumpUpWeakColumn. We are predicting that there is no weak column for boosting.

**4.Boost Factor:**

I did two unit tests to analyze boost factor. Every Input in these unit tests are internally run 3 times by a loop inside this experiment and so is called step 0, step 1, step 2. The two Unit tests are:

a) CheckBoostFactorWithOutUpdate()

b) CheckBoostFactorWithUpdate()

The scenario for this testing is to figure out if MAX\_BOOST parameter have effects on the performance of spatial pooler.“MAX\_BOOST” is a maximum number of boost factor. This parameter is used to boost the overlap values of columns. The boost factors are associated with each column. Each column has a boost factor, and it is just a multiplier for the overlap score of that column. So, before we move to the inhibition phase, we can affect which column end up learning and which one end up expressing themselves and which one do not express themselves and that is what boosting does. This is useful because it helps inactive columns to have a chance to “express” themselves and become more competitive with other columns. Each column's overlap gets multiplied by a boost factor before it gets considered for inhibition. The actual boost factor for a column is number between 1.0 and “max boost.”

There are several inputs coming from encoder to spatial pooler. In our case, we took only 10 inputs (value ranges from 0 to 9). Here, we have analyzed the effect of boost factor.

1. **CheckBoostFactorWithOutUpdate()**

In this experiment, When boosting is off (When boost factor is at minimum level of 1 ) there may be small minority of columns that are even contributing to the model at all but the chance is almost none.

Fig 6. input 1, step 0

1. **CheckBoostFactorWithUpdate()**

In this experiment, The MAX\_BOOST is set 5 here. MAX\_BOOST is a feel that how aggressive our boosting algorithm is going to be to do the homeostatic normalization. When boosting is on, the spatial pooler is much more efficient because it uses more of its columns to represent the data/to model this data as we see in fig 7. So, when we have boosting on, we can spread that around and take advantage of all those columns that could potentially be contributing to the model. Boosting encourages the less active columns become more active. It is very important mechanism in spatial pooler. Otherwise, we cannot get that efficiency what we need. Here is the output representation of input 1 step 0 and input 1 step 2.

**Input 1 Step 0**

When loop is run for the first-time columns gets active and is potentially contributing to the model.

Fig.7 Input 1, step 0

**Input 1 Step 2**

When loop is run for the third time more and more columns get active and taking advantage of all those columns that could potentially contributing to the model.

Fig.8 Input 1, step 2

**5.Inhibition Radius:**

if (IsUpdateRound(this.connections))

{

UpdateInhibitionRadius(this.connections);

UpdateMinDutyCycles(this.connections);

}

}

Here, IsUpdateRound(this.connections) is a Boolean function. If it is true, then we will update the

1. Inhibition Radius
2. Min Duty Cycle

IsUpdateRound(this.connections) is a Boolean function. If it is true, then the learning process will happen. In order to do perform the ‘learning’, the permanence values of the synapses have to be updated and hence the ‘state’ of the model will be modified.

Inhibition is the nameof the mechanism which maintains the sparse activation of the columns [3]. It means, in SP, columns inhibit nearby columns from becoming active. So, inhibition radius is the size of a column’s local neighbourhood, within which the columns inhibit each other from becoming active.

In spatial pooling, if the top k mini columns are activated, then inhibition occurs. It can be called a competition between all the mini columns in a neighbourhood.

The inhibition radius determines the size of a column’s local neighbourhood. A cortical column has to surpass the overlap value of columns in its neighbourhood to become active. The inhibition radius is updated every learning round. It increases and decreases with the average number of connected synapses per column.

There are two types of inhibition. they are [4].

1. Local inhibition: If there are many neighbours and they are distributed throughout the layer, it is called local inhibition. It is more complex, and it assumes topology in the input data.

2. Global inhibition: when there is only one neighbourhood, then all the mini columns compete against each other. This is called global inhibition.

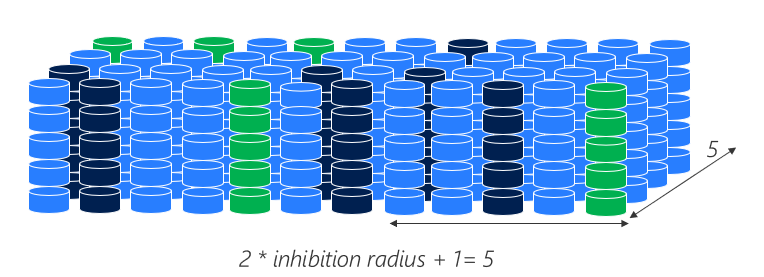
[](file:///C:\Users\49176\Desktop\ll.png)

Fig 9: Global inhibition

**6.Minimum Duty Cycle:** Generally, duty cycle is the fraction of time in which the signal is active. In HTM spatial pooling, it means a time duration cycle that notices which column is active for which time after inhibition. This is also called active-duty cycle which call as a function. It updates the minimum duty cycle in a global way.

Before the inhibition, if a column’s overlap duty cycle is less than the acceptable value calculated dynamically as a function of minOverlapDutyCycle and the overlap Duty cycle in the neighbouring columns, then by the increment amount of all its permanence values are boosted.

|  |  |  |  |
| --- | --- | --- | --- |
| Input | Step | Min Active-Duty Cycle | Min Overlap Duty Cycle |
| 0 | 0 | 0.00002 | 0.00002 |
| 1 | 0.0000396 | 0.0000396 |
| 2 | 0.000038808 | 0.000058808 |
| 1 | 0 | 0.000038031 | 0.000077631 |
| 1 | 0.000037271 | 0.000096079 |
| 2 | 0.000036525 | 0.00011415 |
| 2 | 0 | 0.00003579 | 0.000131874 |
| 1 | 0.000035079 | 0.00014923 |
| 2 | 0.000034377 | 0.000166 |

I got the above values from the unit test. Here is the minimum active cycle and minimum Overlap duty cycle values are increasing so that cells are more active than before.

1. **CONCLUSION**

In this section, you summarize your main findings, comment on those findings (see below), and connect them to other research. You also discuss limitations of your study, and use these limitations as reasons to suggest additional, future research.

This is some not sufficient example. Based on the above experiments, it can be said that various parameters of Boosting algorithm have effect on learning. I have clearly observed that parameters like permanence, boost factor, Active duty cycle, minimum duty cycles, overlap duty cycles, inhibition radius affect the learning by changing the values of these parameters. I have tested these parameters by making unit tests for each method in boosting algorithm and the results were documented.

This is an example of the well written conclusion:  
The Hierarchical Temporal Memory algorithm isinspired by the neo-cortex and implements manyknown features that have roots in neuro-sciences.Nowadays many results show that the algorithm isvery flexible and can solve different kind ofproblems. However, the reverse engineering of theneo-cortex is still a complex and unsolved task. Manydesign decisions in the algorithm base on assumptionsand work in progress. This paper focuses on theinstability issue of the HTM Spatial Pooler algorithm,which has a task to memorize spatial patterns in anunsupervised way. As discussed, the original SpatialPooler already integrates some sort of homeostaticplasticity mechanism discovered in previous work inneurosciences. However, the existing solution causesinstability in the learning process, which makes verydifficult to build applications. This work brieflydocumented the named issue and offered the solutionby extending the existing SP algorithm with the newcomponent called Homeostatic Plasticity Controller.The extended version of the SP is motivated byfinding in neurosciences, that documents the activityof this mechanism during the development of thespecies. Inspired with this finding the newHomeostatic Plasticity Controller defines the

*newborn*

 stage of the Spatial Pooler. In this stage, theSP stimulates the boosting of mini-columns and firstallows the instability in the learning process. Afterthe specified number of iterations, the HPC switchesoff the boosting and waits for the SP to enter thestable state. With this approach the SP converges tothe stable state and applications can be notified aboutthe state of the SP. This improves the quality of thelearning of the SP and enables the implementation ofmore reliable solutions. Another work in progress inthis context is related to the design of the parallelversion of the HTM. The new HPC algorithm needsto be validated for parallel implementation (Dobric,Pech, Ghita and Wennekers, 2019).

**REFERENCES**

[1]. Hole K.J. (2016) The HTM Learning Algorithm. In: Anti-fragile ICT Systems. Simula SpringerBriefs on Computing, vol 1. Springer, Cham. https://doi.org/10.1007/978-3-319-30070-2\_11

[2]. Hole, K. J. (2016). *Anti-fragile ICT Systems.* doi:10.1007/978-3-319-30070-2

[3]. <https://numenta.com/resources/biological-and-machine-intelligence/spatial-pooling-algorithm/>

[4]. Mnatzaganian, J., Fokoué, E., & Kudithipudi, D. (2017). A Mathematical Formalization of Hierarchical Temporal Memory’s Spatial Pooler. Frontiers in Robotics and AI, 3. doi:10.3389/frobt.2016.00081

[5]. Pech, Andreas & Ghita, B.V. & Wennekers, Thomas. (2021). Improved HTM Spatial Pooler with Homeostatic Plasticity Control. 98-106. 10.5220/0010314200980106.

[6].https://numenta.com/resources/biological-and-machine-intelligence/spatial-pooling-algorithm/

..