A GPU based virtual screening tool using SOM

P.B. Jayaraj*, K.M. Mithun and G. Gopakumar

Department of Computer Science and Engineering,
NIT Calicut,
673601, India
Email: jayarajpb@nitc.ac.in
Email: mithuncsnit@gmail.com
Email: gopakumarg@nitc.ac.in
*Corresponding author

U.C.A. Jaleel

OSPF NIAS Drug Discovery Lab,
NIAS,
IISc Campus, Bangalore, 560012, India
Email: jaleel.uc@gmail.com

Abstract: This paper attempts to introduce the applicability of low cost graphics processing unit alternatives to a virtual screening technique using a novel self-organising map (SOM) based technique. This method combines the unsupervised learning capability of the SOM with a subsequent supervised labelling of the trained SOM neurons for building the prediction model. This novel iteration-based SOM technique can label molecule as undefined classes which can reduce the false positives in the screening. For running large datasets, the serial implementation of the proposed algorithm is very time-consuming and cannot be completed in a stipulated time frame. This has been overcome by exploiting the parallelism present in finding the winner neuron and neuron weight updating steps. A tool named SOMSCREEN is developed based on the proposed parallelised method to make the drug discovery process faster. It is observed that, the proposed method offers reduced false positive rate than the Random Forest based work.

Keywords: ligand based drug design; ANN; artificial neural network; virtual screening; self organising map; graphics processing unit; CUDA; parallel computing; winner neuron; bio-assay; unsupervised learning; neighbourhood.

Reference to this paper should be made as follows: Jayaraj, P.B., Mithun, K.M., Gopakumar, G. and Jaleel, U.C.A. (2021) 'A GPU based virtual screening tool using SOM', *Int. J. Computational Biology and Drug Design*, Vol. 14, No. 1, pp.64–80.

Biographical notes: P.B. Jayaraj received his PhD in Computer Science from National Institute of Technology Calicut, India. Now, he is an Assistant Professor at the Department of Computer Science and Engineering, NIT Calicut, India. His research interests include healthcare, computational drug design and artificial

intelligence. He is author of a great deal of research studies published at national and international journals as well as conference proceedings.

K.M. Mithun received his BTech in Computer Science and Engineering from NIT Calicut. His research interest during undergraduate studies was on GPU computing. He is currently working as a Software Engineer at Mathworks, Hyderabad, India.

G. Gopakumar received his PhD in Bioinformatics from the University of Kerala, India. Currently he is working as an Assistant Professor with the Department of Computer Science and Engineering, National Institute of Technology Calicut, India. His research interests include RNA Bioinformatics, Biological Network Analysis, and Machine Learning. He is a Member of the IEEE and ACM.

U.C.A. Jaleel is a Principal Scientist at the OSPF-NIAS Drug Discovery Lab, NIAS IISc Campus Bangalore. He leads a team of drug discovery researchers who focus on various aspects of cheminformatics including AI-based analysis of neglected tropical diseases like TB, Malaria.

1 Introduction

Ligand based virtual screening builds a conceptual model of the target using the data obtained from wet lab experiments (Ripphausen et al., 2011). This model can be viewed as a two dimensional matrix in which the rows correspond to the ligands and the columns represent the features or descriptors of the ligands.

Each set of ligand interaction data contains a large number of molecules as input. From a machine learning perspective, the input data molecules can be viewed as training samples with the molecular descriptors as features of each such training sample (Ekins et al., 2007). Hence, the input to a machine learning algorithm is a large number of molecules labelled active or inactive with their features as training samples. By applying effective classification algorithms on this data, the activity of molecules with the target protein can be studied (Schirez, 2009). This algorithm can learn from these training samples and can create a model, using which it can predict molecules whose interaction with the target have not yet been studied using wet lab experiments (Senanayake et al., 2013). This avoids the necessity to conduct wet lab experiments on a large number of molecules that have been screened as inactive by the classification algorithm (Chen et al., 2007).

Different virtual screening techniques have been developed using machine learning techniques like artificial neural networks (ANN) (Unterthiner et al., 2005), support vector machines (SVM) (Burbidge et al., 2007), random forest (RFC); Jayaraj et al. (2016) and Naive Bayes classifiers (NBC) (Alpaydin, 2003). But self-organising map (SOM), a type of ANN has had limited uses in ligand based virtual screening (Kohonen, 2005). Although SOM is trained through unsupervised learning, it can also be used to build supervised prediction models. This is required in situations when the inherent pattern of the input data needs to be processed first through unsupervised learning before the test data is used in supervised learning. This principle has been used in this work to develop an iterative semi-supervised learning algorithm for virtual screening using SOM. The cost effectiveness and low power-consumption of heterogeneous computing using GPUs make it highly attractive to use them to implement the proposed algorithm in this work (Kirk et al., 2010).

The rest of the paper is organised as follows. Section 2 provides the state of the art details of SOM. Section 3 explains the proposed work in detail. Section 4 gives the experimental evaluation, results and discussion. Finally Section 5 concludes the work.

2 State-of-the-art

2.1 Self-organising map (SOM): methodology

Self-organising map is an artificial neural network (ANN), implemented using an unsupervised system of competitive learning technique (Kohonen, 1990). Competitive learning involves the output neurons competing among themselves to be activated, resulting in only one being activated at any given singular instance of time. The aim of a SOM is to transform an input pattern of multiple dimensions to a two-dimensional discrete map, usually as a square or hexagonal grid. SOM can recognise inputs never encountered before and labels the new unidentified inputs. This subsection describes the working of SOM in brief.

The network architecture of SOM and its Input-Output training details are shown below (Figure 1).

Input: Training set of p distinct vectors X_1 , X_2 , ..., X_j ..., X_p , each vector is of length n and its components are rational numbers;Fausett (1993).
$X_1 = (x_{1,1} , x_{1,2} , \ldots, x_{1,i} , \ldots, x_{1,n})$
$\ldots$
$X_j = (x_{j,1} , x_{j,2} , \ldots, x_{j,i} , \ldots, x_{j,n})$
$\ldots$
$X_p = (x_{p,1} , x_{p,2} , \ldots, x_{p,i} , \ldots, x_{p,n})$

Output: m SOM neurons represented by a vector Y : $(Y_1 , Y_2 , \ldots, Y_i , \ldots, Y_m)$;

Each vector from the training set falls in one of the m SOM neurons.

Algorithm 1 is the general algorithm for training a SOM. The algorithm begins with map initialisation by random values. Each neuron will be assigned with a vector which has the same number of dimensions as the vectors in the input set. This vector will be termed as the weight of the neuron. From the input set, the first input vector is selected and a discriminant function is used to determine the neuron closer to this input vector. This causes a competition between the neurons, out of which only one is chosen. This neuron is the winning neuron, also known as the best matching unit (BMU). When one neuron is fired, the nearest neighbours are affected more than the others. This requires calculating the radius of the neighbourhood. Figure 2 shows how the neighbourhood region forms around a winner neuron in a SOM. The topological neighbourhood decreases exponentially with distance. The winner neuron is then updated and brought closer to the selected input neuron. The other neurons which are away from the winner neuron are also updated. This weight updation will be decreasing exponentially as the distance from the winner increases. It is computed on the basis of their position in the topographical neighbourhood. This process continues till the neuron set becomes stable. The neighbourhood size and the learning rate of every neuron in the map will be maximal at the beginning and will diminish periodically as the algorithm advances.

Figure 1 SOM network architecture

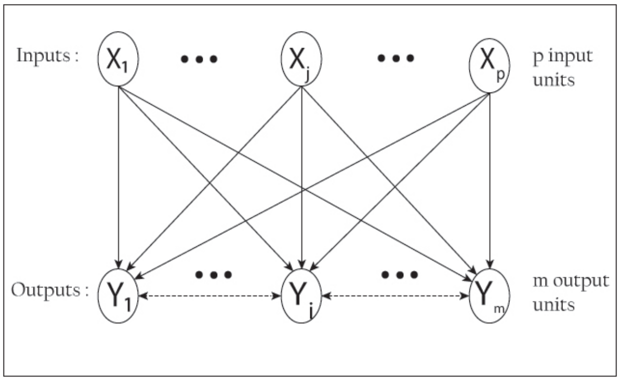
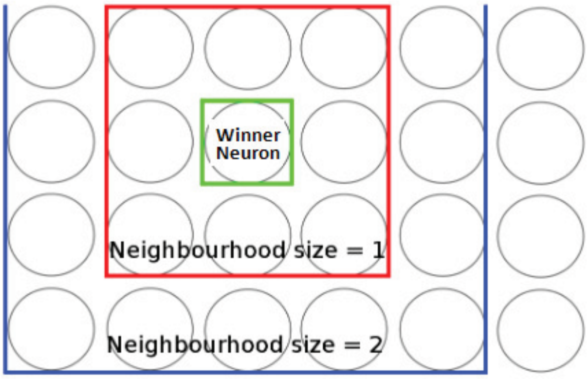


Figure 2 SOM network architecture



2.2 *SOM – Related works*

The proposal of SOM was done by Kohonen (2000) for implementing a document classification system mainly. This section briefs some of the existing works in virtual screening using SOM principles. Kohonen (1990), in his work gives an in-depth explanation about SOM, it's various properties and the method to train a SOM. Selzer and Ertl (1996) in their work combined SOM and radial function with molecule descriptors for biological application. Hristozov et al. (2007) described the techniques for similarity based ligand based virtual screening in which SOM is used as a novelty detection device. Rudolf Mayer et al. (2007) compared different SOM's trained by the same training dataset using different parameters or initialisations.

Algorithm 1 Self Organising Map - Serial Algorithm ;Laurene V. Fausett (1993)

```
 1: function Main
 2:     $map \leftarrow$ Initialise()
 3:     while CONVERGENT(map)=False do
 4:         $currentInputVector \leftarrow$ GetInput()
 5:         $winner \leftarrow$ FindWinner($map, currentInputVector$)
 6:         UpdateMap(map, winner)
 7:         if $currentIteration$ mod 100 == 0 then
 8:             ReduceNeighbourhood
 9:             ReduceLearningRate
10:     return map

11: function FindWinner(map,inputVector)
12:     $minDistance \leftarrow \infty$
13:     for all neurons in map do
14:         $distance \leftarrow$ EuclideanDistance($neuron, inputVector$)
15:         if $distance < minDistance$ then
16:             $winner \leftarrow neuron$
17:             $minDistance \leftarrow distance$
18:     return winner

19: function UpdateMap(map,winner)
20:     for all neurons in map
21:         $neighbourhoodCoeff \leftarrow$NeighbouroodFunction($winner, neuron$)
22:         UpdateNeuron(neuron,neighbourhoodCoeff,winner)
```

In another work, Roche et al. (2002a) developed a tool using SOM for the identification of potential frequent hitters. For the substructure analysis, a scoring scheme was developed using a multivariate linear and nonlinear statistical methods. SOM approach was used to evaluate the usefulness of molecular descriptor sets. Using SOM's high dimensional description space, it can easily identify if the nonlinear projection gives a frequent hitter or non frequent molecule clusters. Roche et al. (2002b) have developed an *in-silico* method for predicting the affinity of low molecular weight compounds. SOM was one of the technique applied to identify the molecular descriptors of these molecules to create structure activity relationship models.

Cross-linking mass spectrometry is largely utilised for structural characterisation of multi-subunit protein complexes. Ferber et al. (2016) introduced an automated modelling method for large protein assembliotgees using cross links distance restraints. Zell et al. (1994) developed self-organising surface, an extension of the SOM, to cope with the typical topological defects that SOMs develop when mapping the surface of a 3D object to a 2D torus of neurons. The similarity analysis of molecules was achieved using the above developed Self organising surface. Bouvier et al. (2015) developed an automatic tool using SOM for clustering macro molecular conformations. They have developed a python library implementing the full SOM analysis work-flow. They also coupled this with visualisation tools and have got Interesting properties.

There have been very few works found in the literature discussing the usage of SOM for machine learning based ligand based virtual screening process. The pattern fetching capability of SOMs have not been explored to the fullest for ligand based virtual screening purpose so far. Running the classical SOM algorithm serially even on a powerful computer with large data inputs cannot complete execution in a limited time frame (Kohonen, 2000). However, the inherent parallelism in the SOM algorithm makes it suitable to implement it on many-core computers. These factors motivated us to propose the current work using GPUs (low cost alternative to supercomputers).

2.3 Existing Parallel SOM Implementations

Many different strategies for parallelising SOM algorithm are available in the literature. These are primarily achieved through either neuron level parallelism or training sample parallelism. Raghavendra D Prabhu (2008) in his work "SOMGPU", implemented a GPU based SOM using non-CUDA GPGPU technique. The crux of the work is processing more than one input vector at a time. Myklebust and Solheim (1995) put forth ideas on node parallelism and training sample parallelism in SOM. The work by McConnell et al. (2012) compares different parallel SOM implementations using OpenCL, CUDA and MPI, which are frameworks for different multi core architecture. Wittek et al. (2015) invented a massively parallel tool for training SOMs, "Somoclu", for large datasets. Davidson (2015) proposed a parallel version of the original SOM algorithm using OpenCL.

3 Method

A new iterative SOM based classifier algorithm is proposed here for virtual screening the ligand data. An iteration of the algorithm consists of model building phase followed by a prediction phase. In the model building phase, SOM and its parameters are initialised with respect to the given training set. Self organising map is then trained with molecules from the training set using unsupervised learning. The trained SOM then undergoes a supervised labelling process. Each SOM neuron is labelled as **a**, **i**, or **nl** depending on whether large number of actives, or large number of inactives, or nearly equal number of actives and inactives respectively falls in that particular SOM neuron cluster. The labelled SOM obtained is our prediction model used in prediction phase. In the prediction phase, for each molecule in test set whose activity needs to be predicted, a corresponding winner SOM neuron is found based on a distance function the Euclidean distance. Those molecules that have winner SOM neurons labelled as **a** or **i** are predicted as active or inactive respectively. Figure 3 illustrates the block representation of the proposed method.

During the model building phase, some molecules from the training set could have resulted in forming **nl** labelled SOM neurons. Such SOM neurons are formed because this method was ineffective in classifying them as active or inactive. The causes affecting formation of **nl** neurons are the influence exerted by molecules in **a** and **i** labelled clusters and the initial parameters for the neurons. These problems are solved by successive iterations of model building phase and prediction phase using the active and inactive molecules from the **nl** labelled SOM neurons obtained from the previous iteration (See Algorithm 2).

Molecules in the test set that have **nl** labelled SOM neurons as winner neurons in previous iterations are sent back to next iteration of the prediction phase. In principle, successive iterations builds a better prediction model for test data that could not be classified appropriately in the previous iterations. The iterations are continued till all molecules of the test set are predicted as active, inactive or unlabelled. When ran with larger datasets, the serial implementation cannot complete in a given time frame because of the costly winner neuron finding and weight updating steps. Therefore a parallel algorithm using GPU was implemented which can do the winner neuron finding and weight updating steps in a faster manner. Algorithm 3 shows the parallel algorithm used in this process. The serial iterative algorithm narrated in Algorithm 2 is rewritten for exploiting the parallelism in it. The map initialisation is also be done in a parallel way.

Algorithm 2 Proposed serial iterative SOM based algorithm for Virtual Screening

```
    Input
    $Trset$ - Matrix of features of the training ligand data set(actives and inactives)
    $Tset$ - Matrix of features of the test ligand data set
    Output
    Tset - Activity of the ligands whose features are specified in Tset
 1: Fix the network topology as square grid for SOM neurons.
 2: Repeat following steps until either training set is exhausted or all the molecules in the
    test set are predicted as $a$, $i$ or **u**.
 3: Initialize the SOM with numberOfSOMneurons = $4 \times \lfloor 5 \times \sqrt{z} \rfloor$ (J Vesanto et al. (2000)).
    Where z is the no of molecules in the training set.
 4: Initialize weights of the SOM neurons to random molecule descriptors picked from
    Trset.
 5: Initially label all SOM neurons as $u$.
 6: Repeat steps 7 to 11 until the Trset is completely used up.
 7: Choose a molecule **k** at random from training set.
 8: Compute the euclidean distance of molecule k from each of the SOM neuron.
 9: Choose winner SOM neuron as the one having minimum distance function from k. It is
    also called Best Matching Unit (BMU). Save the BMU as **k**'s winner SOM neuron.
10: Calculate radius of neighborhood of BMU neuron.
11: Update weight of BMU and the SOM neurons within the radius of neighborhood.
    The weight update function should comprise of learning rate and distance from BMU
    parameters.
12: Label SOM neurons as:
13:     **a**: if $\%$ of active molecules from $Trset$ that fall in the SOM neuron is greater than 90.
14:     **i**: if $\%$ of active molecules from $Trset$ that fall in the SOM neuron is less than 10.
15:     **nl**: if $\%$ of active molecules from $Trset$ that fall in the SOM neuron is in between 90
    and 10.
16: All molecules from Trset that fell in SOM neurons labelled as *nl* are kept for the next
    iteration. The remaining are discarded.
17: Repeat steps 18 to 19 until test file is completely perdicted out.
18: Choose a molecule **k** from Tset.
19: Find BMU for **k**.
20: Discard and save the molecule and its BMU label if its BMU is labelled as $\boldsymbol{a}$, $\boldsymbol{i}$ or $\boldsymbol{u}$.
21: Send remaining molecules in Tset that have neurons labelled as **nl** for next iteration.
```

Algorithm 3 Proposed Parallel iterative SOM based algorithm for Virtual Screening

```
Input
$Trset$ - Matrix of features of the training ligand data set(actives and inactives)
$Tset$ - Matrix of features of the test ligand data set
Output
$Tset$ - Activity of the ligands whose features are specified in Tset
function Main
    repeat
        $m \leftarrow NumberOfElements(Trset)$
        $n \leftarrow 4 \times 5 \times \sqrt{(m)}$
        $map \leftarrow$ Initialise()
        for($i \leftarrow 1$ to $m$) do
            currentInputVector ← GetInput()
            $winner \leftarrow$ FindWinner($map, currentInputVector$)
            if $currentInputVector.Label$ ==ACTIVE then
                NeuronActiveMolCount+=1
            else
                NeuronInactiveMolCount+=1
            end if
            UpdateMap(map, winner)
            if $currentIteration$ mod 10 = 0 then
                ReduceNeighbourhood
                ReduceLearningRate
            end if
        end for
        ApplyLabelling(map)
        PredictMolecule(map, TrSet)
        TrSet ← TrSet - 'a','i' Labelled Molecules
        TSet ← TSet - 'a','i' Labelled Molecules
    until (Trset is completely used up OR Tset is fully predicted)

function FindWinner(map,inputVector)
    $minDistance \leftarrow \infty$
    for all neurons in map in parallel do
        $distance \leftarrow$ EuclideanDistance($neuron, inputVector$)
        if $distance < minDistance$ then
            $winner \leftarrow neuron$
            $minDistance \leftarrow distance$
        return winner
    end for

function UpdateMap(map,winner)
    for all neurons in map in parallel do
        $neighbourhoodCoeff \leftarrow$NeighbouroodFunction($winner, neuron$)
        UpdateNeuron(neuron,neighbourhoodCoeff,winner)
    end for
```

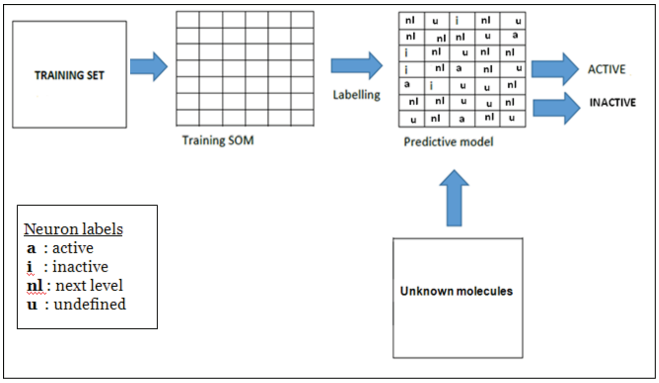
```
function PredictMolecule(map,Tset)
    $m \leftarrow NumberOfElements(Tset)$
    for($i \leftarrow 1$ to $m$) do
        $currentInputVector \leftarrow$ GetInput()
        $winner \leftarrow$ FindWinner($map, currentInputVector$)
        $currentInputVector.Label \leftarrow Winner.NeuronLabel$
        $LabelStore \leftarrow$ currentInputVector.Label
    end for

function ApplyLabelling(map)
    for all neurons in map in parallel do
        $Na \leftarrow NeuronActiveMolCount$
        $Ni \leftarrow NeuronActiveMolCount$
        $Nt \leftarrow Na + Ni$
        $LabelAccuracy \leftarrow CalculateLabelAccuracyLevel$
        if $t > 0$ then
            if $(Na \times 100) \div Nt > LabelAccuracy$ then
                $NeuronLabel \leftarrow$ 'a'
            elsif$((Ni) \times 100) \div Nt > labelaccuracy$
                $NeuronLabel \leftarrow$ 'i'
            else
                $NeuronLabel \leftarrow$ 'nl'
            end if
        end if
    end for
```

FindWinner() is the function used in the Algorithm 3 to find the winner neuron. The euclidean distance from the given input vector to all neurons present in the map is calculated in this function. These distance finding calculations are time consuming but they are also independent in operations. Hence these can be calculated parallely by spawning multiple threads on GPU. Similarly, the function *UpdateMap()* updates the charges of neurons appearing in the neighbourhood of the winner neuron is also calculated independently. The function *PredictMolecule()* predicts the activity of the unknown molecule by finding the winner neuron and assigning it to the same cluster. *ApplyLabelling()* function gives the suitable label to the clusters by seeing the percentage of active-inactive shares in it. The *main()* function coordinates the all the above said functions to enable SOM for virtual screening activity.

4 Experimental evaluations

The computational performance and efficiency measures obtained for the proposed method are described in this section. The details of datasets used in the implementation and the parameters used in SOM creation are also explained in this section.

Figure 3 SOM based virtual screening: a block representation (see online version for colours)



4.1 *Datasets used*

Datasets with different number of molecules were used to gain insight into the quality of the result produced by this GPU version. Training datasets were obtained from NCBI (PubChem, https://pubchem.ncbi.nlm.nih.gov/) bioassay database which was prepared from frozen stocks of Mtb H37Rv obtained from American Type culture collections. For the screen, amikacin was included in the positive control wells in every assay plate. The datasets used for training the models as shown in Table 1. The bio-assay SDF file downloaded from PubChem was supplied as input to the (PowerMv, http://nisla05.niss.org/PowerMV/) feature extraction tool to generate 2D molecular descriptors. A total of 179 descriptors were generated for each dataset of the experiment. The selection of descriptors was based on the criteria that they are sufficient to characterise the drug-likeness of a compound; Schirez (2009). These descriptors fall into three categories. The first eight descriptors are used mainly to characterise the drug-likeness of a compound. Another set of twenty-four continuous descriptors considered are based on a variation of BCUT descriptors to define a low dimensional chemistry space. The last 147 bit-string structural descriptors, known as Pharmacophore Fingerprints, are based on bioisosteric principles. The preprocessed data can be visualised as a two-dimensional matrix in which the rows correspond to the ligands and the columns represent molecular descriptors of a ligand. The matrix entries are generally real numbers based on the type of descriptors selected. Typical molecular descriptors used are molecular weight, pH value, blood brain barrier and polar surface area. Higher precision and accuracy can be obtained by selecting more relevant descriptors for screening. This training matrix also carries the class label information of each molecule as either active or inactive. The PowerMV feature extracted matrix data, usually stored as text file, will be supplied to the proposed SOM classifier for training purpose. The dataset for testing is taken from GDB17, a chemical universal database for unknown compounds which has been enumerated by Lars et al. (2012). Compared to the 2.5 million known molecules found in PubChem, the GDB-17 database has 166.4 billion molecules. The other databases such as GDB-11 and GDB-13 only address very small organic molecules, which are of relatively small size. GDB-17 molecules contain generally more rings, in particular small rings, as well as many nonaromatic heterocycles.

Table 1 Datasets used for training

PubChem Bioassay Datasets	*Number of Molecules*	*Date of access*
AID 1332	1093	26/07/2020
AID 492952	2294	26/07/2020
AID 651616	5569	26/07/2020
AID 2330	36869	26/07/2020
AID 2842	23462	26/07/2020

AID stands for Assay ID. Assay is an investigative analytic procedure in pharmacology for quantitatively measuring the functional activity of a target entity.

4.2 Hardware used

A server having two Xeon 4116 CPU with a 96GB RAM and a V100 GPU card containing 32 GB Graphics RAM is used for experimentation. The detailed specification of the machine used is shown in Table2.

Table 2 Hardware configuration used

Particulars	*GPU*
GPU	Tesla V100
CUDA Cores	5120
GPU Clock Speed	877 MHz
Graphic Memory	32 GB
Memory Bandwidth	900 GB/Sec
Compute Capability	7.0
CPU	Xeon 4116 (12 core) X 2
CUDA version	11.0

4.3 SOM parameters used

The network topology used here for the SOM is a square grid. Minimum size of square grid is taken as 3×3. The dimension of the SOM grid can be expressed as $\sqrt{y} \times \sqrt{y}$, where y is the number of SOM neurons.

The functions used as the SOM parameters in the algorithm implementation are shown below (Buckland, 2005)

Weight Update Function: $W(t+1) = W(t) + \Theta(t)L(t)(I(t) - W(t))$

Learning Rate: $L(t) = L_0 \exp^{-t/\lambda}$

Radius of neighbourhood: $\sigma(t) = \sigma_0 \exp^{-t/\lambda}$

Distance from BMU: $\Theta(t) = \exp^{-distancefunctionfromBMU/2\sigma^2(t)}$

I = current input, W = SOM neuron's weight, t = current iteration.

4.4 Results

Table 3 shows the efficiency measures of the serial SOM algorithm for different datasets in terms of accuracy, precision, recall and F-score (Schirez, 2009) on various bioassay datasets.

Table 4 depicts the same details for GPU parallel algorithm. From these two tables, it is clear that the efficiency is not affected by parallelising the finding winner neuron and weight updation of neighbourhood steps. Ten-fold cross validation techniques are applied to bench mark the performance metrices.

Table 3 Performance measures of serial SOM algorithm for different datasets

Dataset	*AID 1332*	*AID 492952*	*AID 651616*	*AID 2330*	*AID 2842*
Accuracy	0.84	0.67	0.72	0.91	0.94
Precision	0.48	0.74	0.82	0.42	0.38
Recall	0.41	0.74	0.83	0.33	0.30
F-Score	0.44	0.74	0.82	0.37	0.33

Table 4 Performance measures of GPU parallel SOM algorithm for different datasets

Dataset	*AID 1332*	*AID 492952*	*AID 651616*	*AID 2842*	*AID 2330*
Accuracy	0.86	0.67	0.73	0.92	0.93
Precision	0.53	0.73	0.82	0.43	0.38
Recall	0.47	0.76	0.85	0.342	0.30
F-Score	0.50	0.74	0.84	0.38	0.33

Though efficiency matrices of the SOM algorithm are good, an improvement is observed in the classification phase as execution speedup. The serial version was taking a long running time even in a server machine (which has 96GB RAM) and could not complete the execution in a stipulated time frame, when large data is applied. This is due to the intensive computation present in the winner neuron finding and neuron weight update steps. The comparison of run time for the classification of various size data is shown in Table 5. Evidently, a tremendous speed up is achieved in the execution of this proposed algorithm. This contributes to significant improvement in virtual screening of ligand based data models.

Another advantage of this algorithm is that it can separate the test molecules into *undefined* group. These molecules would have mispredicted as either active or inactive if other supervised classifiers like Random Forest and SVM were used. This is due to the fact that these learning algorithms are binary classifiers and has to place these undefined molecules into either actives or inactives group. Here, SOM can detect the compounds which lie outside the activity space as a separate set. This will reduce false positive rate of the classifier. Table 6 shows the number of actives, inactives and *undefined* compounds predicted in the serial SOM execution. Though the compounds are randomly chosen from the test set, due to the inherent capability of SOM, the best compounds in terms of activity will be detected in the initial levels of iterations.

The proposed SOM based virtual screening method has been compared with a Random Forest (Jayaraj et al., 2016) and SVM based methods; Jayaraj and Jain (2016). The results obtained after executing the two methods using the same dataset, are presented here for comparison.

Table 5 Running time of serial and GPU parallel versions of SOM based virtual screening

Dataset	*No. of molecules (in millions)*	*Exec.Time(sec) GPU parallel*	*Exec.Time(sec) Serial*	*Speedup*
GDB17	0.1	500.34	60003.6	120.2
GDB17	0.3	1091.34	276983.9	131.35
GDB17	0.5	2291.34	558654.1	243.35
GDB17	1	4723.1	1277834.2	270.5
GDB17	2	9706.85	*	#
GDB17	5	24348.05	*	#
GDB17	10	79082.6	*	#

*: Not Executed due to serial exception error, #: large value.

Table 6 Number of active, inactive, *undefined* compounds predicted by GPU SOM based classifier

Testset	*No of molecules* (millions)	*Actives*	*In-actives*	*Undefined*
GDB17	0.5	6089	474961	17950
GDB17	1	17439	945046	37415
GDB17	2	30524	1912083	57393
GDB17	5	87631	4785065	127304
GDB17	10	183474	9419674	396852

Table 7 shows the performance comparison of principle component analysis (PCA) applied parallel RF and SOM based VS methods. The input to the Parallel RF was PCA applied. From the comparison, it can be inferred that the efficiency matrices of SOM classifier is at par with RF Classifier. But the main advantage of the SOM classifier is, it can label molecules as *undefined* other than actives and inactives. This can reduce the number of false positives that may have results in the other classifiers. These molecules can be send to medicinal chemist for detailed study. The identification of this undefined compounds make SOM classification better than RFC for Virtual screening the ligand molecules.

Table 7 Performance comparison of efficiency metrics of Random Forest and SOM classifier for virtual screening

Dataset	*AID 1332*	*AID 492952*	*AID 651616*	*AID 2330*
Efficiency matrix – Parallel RF Classifier				
Recall	0.61	0.71	0.77	0.49
Precision	0.46	0.84	0.95	0.33
F-Score	0.53	0.76	0.85	0.39
Accuracy	0.85	0.67	0.75	0.92
Efficiency matrix – Parallel SOM Classifier				
Recall	0.47	0.76	0.85	0.30
Precision	0.53	0.73	0.82	0.36
F-Score	0.50	0.74	0.84	0.33
Accuracy	0.86	0.67	0.73	0.93
Undefined Compounds (0.5 millions dataset, GDB17)	30678	25453	23567	17950

This advantage is more visible in Table 8 when the no of actives predicted by SVM classifier was compared with SOM based method. A Directory of Useful Decoys - DUD datasets (Niu et al., 2006) was used for comparing the above classification results. The most no of actives predicted by SVM is classified as undefined in SOM method. It can be inferred that these set of molecules are wrongly predicted as actives in SVM method.

Table 8 Classification analysis (no of actives and undefined compounds classified) of Parallel SOM & SVM Classifiers on DUD molecules using AID2330 as training model

LigandSet	*#Molecules*	*Parallel Exec. Time*	*No. of actives in SVM classifier*	*No. of Actives in SOM classifier*	*No. of undefined compounds in SOM*
ache_decoys	3892	173.03 s	743	285	426
egfr_ligands	475	176.88 s	19	25	38
dhfr_ligands	410	138.62 s	27	20	30
cdk2_ligands	71	141.14 s	50	2	0
ar_ligands	79	187.85 s	11	1	9
Thrombin_ ligands	72	138.76 s	11	2	9

One of the main achievement of our proposed method is the large speed up achieved in GPU parallel SOM classification without any loss of efficiency. Another advantage is the finding of Undefined molecules which can be used by medicinal chemists for making further investigations and inferences. The serial SOM based algorithm designed for this purpose was time taking and could not complete its execution on time when no of molecules for classification grows to tens of millions. When implemented it using GPU kernels, the algorithm could scale to take large input and screen billions of molecules in limited time. We hope , this will be benefited for the fast design of drugs in computational drug discovery. As a work enhancement to achieve a better prediction, other distance measures such as Manhattan distance can also be considered as discriminant function for the winner neuron prediction.

5 Conclusion

Virtual screening is an ever time challenging problem, which has direct impact in the discovery of drug compounds for many diseases. Though we have different techniques available for solving it, most of them are not 100% perfect. Self organisation map, an intelligent neural network based method, which can fetch the patterns present from the dataset, has not been utilised to its fullest in solving the Virtual Screening problem. We have proposed and developed a SOM based classification technique for virtual screening the ligand data. The serial version of the algorithm could not produce output in a limited time when the test data size increases to millions.

Considering the large volume of data involved in the screening, a parallelised version of virtual screening can save a considerable amount of time. We have parallelised the computational intensive sections of find winner neuron and weight updation in neighbourhood, using GPU. The main advantage of our method is, it can effectively classify the compounds as active or inactive within an acceptable time. The other benefits of the work is that it can label compounds from the test set, which are lying outside the known activity space, as *undefined*. If required, these undefined molecules can be send to medicinal chemists for further investigation. Since the algorithm removes these *undefined* compounds from the test set in the initial rounds of iterations, the possibility of wrong prediction is got reduced. Also, the active compounds will be identified in the initial rounds of iterations due to the pattern fetching capability of SOM. By changing the grid size of the SOM, the behaviour of the compounds which are clustering to similar groups can be studied in detail. The source code of this tool is maintained as an open frame work for further and research.

Acknowledgement

We would like to thank Department of Computer Science and Engineering, NIT Calicut for their constant support. We also thank Central Computer Centre, NIT Calicut for providing the GPU infrastructure for the completing the work. We express our gratitude to Open Source Drug Discovery for providing the dataset for the experimentation.

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Competing interests

The authors declare that they have no competing interests.

References

Almendra, V. and Enachescu, D. (2013) ‘Using self-organizing maps for fraud prediction at on-line auction sites’, *Proceedings of the 15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, pp.281–288.

Alpaydin, E. (2003) *Introduction to Machine Learning*, 2nd ed., The MIT Press Cambridge, Massachusetts London, England.

Bouvier, G., Desdouits, N., Ferber, M., Blondel, A. and Nilges, M. (2015) ‘An automatic tool to analyze and cluster macromolecular conformations based on Self-Organizing Maps’. *Bioinformatics*, Vol. 31, No. 19, pp.1490–1492.

Buckland (2005) *Kohonen’s Self Organizing Feature Maps*, http://www.ai-junkie.com/ann/som/som1.html

Burbidge, R., Trotter, M., Buxton, B. and Holden, S. (2001) ‘Drug design by machine learning: support vector machines for pharmaceutical data analysis’, *Computers and Chemistry*, Vol. 26, No. 1, pp.5–14.

Chen, B., Harrison, R.F., Papadatos, G., Willett, P., Wood, D.J., Lewell, X.Q., Greenidge, P. and Stiefl, N. (2007) ‘Evaluation of machine-learning methods for ligand-based virtual screening’, *Journal of Computer Aided Molecular Design*, Vol. 21, pp.53–62.

Davidson, G. (2015) *A Parallel Implementation of the Self Organising Map using OpenCL*, School of Computer Science, University of Glasgow, Thesis.

Ekins, S., Mestres, J. and Testa, B. (2007) ‘in silico pharmacology for drug discovery: methods for virtual ligand screening and profiling’, *British Journal of Pharmacology*, Vol. 152, pp.9–20.

Fausett, L.V. (1993) *Fundamentals of Neural Networks: Architectures, Algorithms And Applications* Pearson Prentice Hall.

Ferber, M., Kosinski, J., Ori, A., Rashid, U.J., Moreno-Morcillo, M., Simon, B., Bouvier, G., Batista, P.R., Miller, C.W., Beck, M. and Nilges, M. (2016) 'Automated structure modeling of large protein assemblies using crosslinks as distance restraints', *Nature Methods*, Vol. 13, No. 6, pp.515–520.

Hastie, T., Tibshirani, R. and Friedman, J. (2008) *The Elements of Statistical Learning Data Mining, Inference and Prediction*, 2nd ed., Springer: Statistics.

Hristozov, D., Oprea, T.I. and Gasteiger, J. (2007) 'Ligand-based virtual screening by novelty detection with self-organizing maps', *Journal of Chemical Information and Modeling*, Vol. 47, No. 6, pp.2044–2062.

Hung, C. and Huang, J-J. (2011) 'Mining rules from one-dimensional self-organizing maps', *Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications*, pp.292–295.

Jayaraj, P.B. and Jain, S. (2019) 'Ligand based virtual screening using SVM on GPU'. *Computational Biology and Chemistry*, Vol. 83, pp.107143.

Jayaraj, P.B., Ajay, M.K., Nufail, M., Gopakumar, G. and Jaleel, U.C.A (2016) 'GPURFSCREEN: a GPU based virtual screening tool using random forest classifier'. *Journal of cheminformatics*, Vol. 8, No. 1, pp.1–13.

Kirk, D.B., Mei, W and Hwu, W. (2010) *Programming massively Parallel Processors- A Hands-on Approach*, *Morgan Kaufmann Publishers Inc*, San Francisco.

Kohonen, T. (1990) 'The self-organizing map', *Proceedings of the IEEE*, Vol. 78, No. 9, pp.1464–1480.

Kohonen, T. (2000) 'Self organization of a massive document collection', *IEEE Transactions on Neural Networks*, Vol. 11, No. 3, pp.574–585.

Kohonen, T. (2005) 'The self-organizing map, descriptor generation, data analysis and hit evaluation', *Journal of Chemical Information and Modelling*, Vol. 45, No. 2, pp.515–522.

Lars, R., van Deursen, R., Blum, L.C and Reymond, J-L (2012) 'Enumeration of 166 Billion Organic Small Molecules in the Chemical Universe Database GDB-17', *J. Chem. Inf. Model*, Vol. 52, pp.2864–2875.

McConnell, S., Sturgeon, R., Henry, G., Mayne, A. and Hurley, R. (2012) 'Scalability of self-organizing maps on a GPU cluster using OpenCL and CUDA', *Journal of Physics Conference Series*, Vol. 341, pp.012–018.

Mitchell, T. (1997) *Machine Learning*, 1st ed., McGraw Hill.

Myklebust, G. and Solheim, J.G. (1995) 'Parallel self-organizing Maps for actual applications', *Proceedings of IEEE International Conference on Neural Networks*, pp.1054–1059.

Niu, H., Shoichet, B.K. and Irwin, J.J. (2013) 'Benchmarking sets for molecular docking', *Journal of medicinal chemistry*, Vol. 49, No. 23, pp.6789–6801.

Prabhu, R.D. (2008) 'SOMGPU: an unsupervised pattern classifier on graphical processing unit', *IEEE World Congress on Computational Intelligence, Evolutionary Computation*, pp.1011–1018.

Ripphausen, P., Nisius, B. and Bajorath, J. (2011) 'State-of-the-art in ligand-based virtual screening', *Drug Discovery Today*, Vol. 16, No. 9, pp.372–376.

Roche, O., Trube, G., Zuegge, J., Pflimlin, P., Alanine, A. and Schneider, G. (2002a) 'A virtual screening method for prediction of the HERG potassium channel liability of compound libraries', *ChemBioChem*, Vol. 3, No. 5, pp.455–459.

Roche, O., Schneider, P., Zuegge, J., Guba, W., Kansy, M., Alanine, A., Bleicher, K., Danel, F., Gutknecht, E.M., Rogers-Evans, and Neidhart, W. (2002) 'Development of a virtual screening method for identification of frequent hitters in compound libraries', *Journal of Medicinal Chemistry*, Vol. 45, No. 1, pp.137–142.

Mayer, R., Neumayer, R., Baum, D. and Rauber, A. (2007) 'Analytic comparison of self-organising maps', *Proceedings of 7th International Workshop on Self-Organizing Maps (WSOM)*, pp.182–190.

Schirez, A.C. (2009) 'Virtual screening of bioassay data', *Journal of Cheminformatics*, Vol. 21, No. 1, pp.1–12.

Selzer, P. and Ertl, P. (2006) 'Applications of self-organizing neural networks in virtual screening and diversity selection', *Journal of Chemical Information and Modeling*, Vol. 46, No. 6, pp.2319–2323.

Senanayake, U., Prabuddha, R. and Ragel, R. (2013) 'Machine learning based search space optimisation for drug discovery', *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, Vol. 13, pp.1–13.

Unterthiner, T., Mayr, A., Klambauer, G., Steijaert, M., Wegner, J.K., Ceulemans, H. and Hochreiter, S. (2014), 'Deep learning as an opportunity in virtual screening', *Advances in Neural Information Processing Systems*, p.27.

Vesanto, J. and Alhoniemi, E. (2000) 'Clustering of the self-organizing map', *IEEE Transactions on Neural Networks*, Vol. 11, No. 3, pp.586–600.

Wittek, P., Gao, S.C., Lim, I.S. and Zhao, L. (2015) *Somoclu: An Efficient Parallel Library for Self-Organizing Maps*, University of Boras, Technical Report.

Zell, A., Bayer, H. and Bauknecht, H. (1994) 'Similarity analysis of molecules with self-organizing surfaces-An extension of the self-organizing map', *Proceedings of IEEE International Conference*, Vol. 2, pp.719–724.

Websites

Chemistry Development Kit, http://cdk.sourceforge.net/

NCBI PubChem, https://pubchem.ncbi.nlm.nih.gov/

PowerMv Molecular Viewer, http://nisla05.niss.org/PowerMV/