Spiral Organiser for Support Vector Machines

Nachiket Namjoshi
Department of Computer Engineering
Marathwada Mitramandal's College of Engineering
Pune, India

Harshal Chaudhari
Department of Computer Engineering
Marathwada Mitramandal's College of Engineering
Pune, India

Rahul Kolhatkar
Department of Computer Engineering
Marathwada Mitramandal's College of Engineering
Pune, India

Himanshu Londhe
Department of Computer Engineering
Marathwada Mitramandal's College of Engineering
Pune, India

***Abstract*—Support Vector Machines (SVM) have being used very frequently in past few decades. However, they can only be used if the data is less scattered. Scatterednes of the dataset can be solved by reorganizing the dataset. This re-organization has to be in-place, *i.e* it does not require any more memory than it already does. An improved method of organising the data is to be implemented so that the data is much easily classified by SVMs. Spiral Organizer changes the position of the data elements, but keeps the orignal links. The orignal links are kept because of data integrity issues as well as for solving data plotting dependencies.**

***Index Terms*—Data Mining, Support Vector Machines**

I. Introduction

In this age of electronics, knowledge is the key to every possible problem may it be in business, research, medicine or philosophy. The basis of the knowledge is data. The data which can be acquired from any source. It can be well formed and consistent, but, on the other hand, it can also be inconsistent and scattered. As we know that SVMs are particularly poor to work with such datasets. However, it is really important that there must be a way for SVMs to handle scattered data. There exists two types of SVMs:

- Linear SVM
- Non-Linear SVM

Linear SVMs are suitable for smaller datasets with less *scatteredness*, but, Non-Linear SVMs can be used to work with particularly larger datasets with increased *scatteredness*. However, Non-Linear SVMs have a disadvantage of time. Linear SVMs provide results in considerably small amount of time, taking *O(n)* as compared to Non-Linear SVMs which take $O(n^3)$. This paper emphasises on the results of Spiral organiser which is a way of organising the data such that the resulting dataset is *classify-able* by Linear SVMs. This was proposed in our previous paper [2] The organisation of the data is to be done before passing it to the SVM which increases over all efficiency by not only increasing the accuracy but also the time taken to perform classification.

II. Background Work

Since Vapnik proposed Support Vector Machines in 1995 [1], there have been multiple studies as well as attempts to increase their efficiency by speeding up the calculations with the help of Graphics Processing Units (GPUs). Several studies have shown that GPU can be used to accelerate computations with the help of parallelism.

As we already know [2], k-NN algorithm, when implemented on GPU, increased the speed of execution. We see speedup of upto 100x speedup on GPU as compared to CPU [3]. Neetu Faujdar and Satya Prakash Ghrera [4] implemented bubble sorting on GPU with 334 GPU cores which sped up the execution and they obtained a speed-up in time complexity from O(n) on CPU to O(1) on GPU.

In past, there have been several studies and experiments on speeding up SVM with the help of GPU. As summarized [2] T. N. Do and V. H. Nguyen in 2008 [5] proposed a novel Algorithm called *Least Squares SVM* (LS-SVM) which eliminates the necessity of solving quadratic equations. Instead, with LS-SVM, a simple Linear equation is to be solved which is faster than standard SVMs. They claim that 6000x speedup can be achieved with LS-SVM. Q. Li, R. Salman and V. Kecman in 2010 [6] have proposed data chunking and data reduction methods to speedup SVMs, and calculated 13x to 52x speed up on GPU. In 2011, Andreas Athanasopoulos, Anastasios Dimou, Vasileios Mezaris, Ioannis Kompatsiaris [7], proposed an algorithm to solve *Kernel Matrix Computations* on GPU using CUDA. On the other hand, Sopya K., Drozda P., Grecki P. in 2012 [8] proposed an implementation of SVM using sequential minimal optimization (SMO) and compressed sparse row (CSR) and calculated a speedup of 6x to 36x on GPU.

In a survey conducted by Yunmei Lu, Yun Zhu, Meng Han, Jing (Selena) He, and *et. al.* [9], they concluded that a GPU-based SVM can achieve speed-up up to *10-fold* or even higher. With the increase in data created by humans, in the coming years, GPU-based SVM will become more popular.

III. Proposed Work

As we know, SVMs in general have certain disadvantages, to overcome the limitations, we proposed Spiral Organizer (SO). SO provides solution to the data scatterness problem. It does so by rearranging the data elements. SO arranges the data in concentric spirals, each circle represents a data element pair. The concentric spirals is divided in 2 halves, each half is one of the binary classes.

A. Datasets

Spiral Organiser (SO) aims towards reducing *scatteredness*, which means the dataset that should be used with SO contains scattered data items. The dataset must also be in compliance with SVM's constraints:

- Dataset must be large
- Dataset must have binary classes

For testing our proposed methodology, SUSY dataset was used. SUSY is a scientific dataset, which has been produced using *Monte Carlo simulations* [10]. This dataset has *5 million* instances, 18 features and binary classes. Constraints stated above are matched by SUSY dataset.

B. Spiral Organiser

Spiral Organiser re-arranges the data in a spiral manner, where each layer of spiral is considered as a circle, where all the layers of the spiral are like concentric circles with each circle divided in 2 halves such that each semi circle represent one class of data. The swapping of the data elements is done by the basis of comparison with the threshold value. This threshold is computed dynamically. (it will change as per the attributes provided to it). The accuracy of the swapping is increased by using the same 2 attributes that are passed to the SVM classifier. This reduces the unwanted swaps and which results in increase in accuracy

Algorithm of Spiral Organiser can be expressed as:

```
$mat \leftarrow convertToMatrix(dataset)$
$meanbad \leftarrow getThreshold(thresholdParameter)$
$flatmat \leftarrow mat.flatten()$
$obj \leftarrow len(dataset)$
$size_flat \leftarrow len(flatmat)$
$mid \leftarrow (int)(size\_flat/2-1)$
$bot \leftarrow size\_flat-1$
for i:0 to mid do
  if flatmat[i][3] > meanBad and flatmat[bot][3] > meanBad then
    continue
  else if flatmat[i][3] <= meanBad and flatmat[bot][3] > meanBad then
    swap(flatmat[i],flatmat[bot])
    $bot \leftarrow bot-1$
    continue
  else if flatmat[i][3] <= meanBad and flatmat[bot][3] > meanBad then
    $bot \leftarrow bot-1$
    $i \leftarrow i-1$
    if i == bot then
      break
    end if
  end if
end for
```

C. Enhanced Spiral Organiser

Spiral Organiser algorithm is designed in such a way that it can be parallellised. Spiral organiser takes dataset as well as feature list on which SVM is run on. However, in the real world, we may never know which featureset can result in maximum efficiency. As a result, we may need to use SVM on every probable permutation of the pairs of features and by extension, Spiral also. We can parallellise Spiral by passing every permutation to different thread.

However, every thread must access to its own copy of the dataset which is not feasible for using GPU as there is a data-transfer latency. Hence, GPU is not usable for this algorithm.

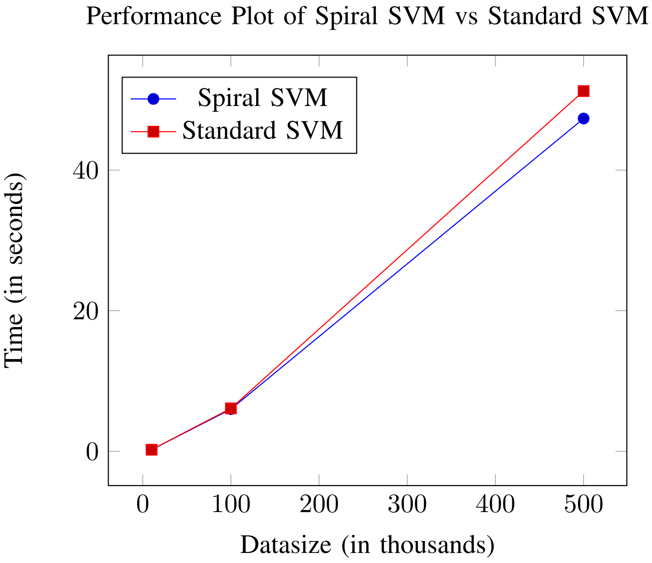
IV. Results

For testing the efficiency, accuracy of classification and total time of execution was calculated using the confusion matrix [11]. Results obtained showed 13-15% increase in accuracy and 2-3% decrease in execution time on CPU. This designed model works efficiently when executed parallely as well. However, the internal of the spiral cannot be parallelized as it requires the entire dataset, to ensure maximum swaps and increased accuracy. Following are the curated results:

TABLE I
Execution Timings

Data Size	**Spiral SVM**	**SVM**	**Percentage Increase (Average Increase in time and accuracy)**
10,000	0.226/87.8%	0.227/74.7%	8.96%
100,000	5.996/89.69%	6.108/75.63%	10.21%
500,000	47.339/89.7%	51.243/75.64%	13.09%

For a better view, when a double bar graph is plotted *(time against data size)*, we get the following plot:



V. Future Work

Although, results obtained on CPU provide an increase in the efficiency of classification. More speed up can be achieved by running the SVM kernel on GPU. Similarly, same can be done in case of SO. During our experiment with the test data, we observed bottleneck in data latency and also GPU parallelization requires to load data onto GPU memory which is not possible with data of such size. One other way discussed earlier [2] is to perform the classification operation on multiple clusters of GPUs.

References

[1] V. Vapnik,“The Nature of Statistical Learning Theory, Springer-Verlag, New York, 1995.

[2] Nachiket Namjoshi, Harshal Chaudhari, Rahul Kolhatkar and Himanshu Londhe. 2018. “Enhanced Support Vector Machine with speed up and reduced sensitivity”, publisher, pagenumber

[3] Selvaluxmiy.S, Kumara.T.N, Keerthanan.P, Velmakivan.R, Ragel.R, Deegalla.S., “Accelerating k-NN classification algorithm using graphics processing units,” 2016 IEEE International Conference on Information and Automation for Sustainability (ICIAfS), Galle, 2016, pp. 1-6.

[4] N. Faujdar and S. P. Ghrera, “A practical approach of GPU bubble sort with CUDA hardware,” 2017 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence, Noida, 2017, pp. 7-12.

[5] T. N. Do and V. H. Nguyen, “A novel speed-up SVM algorithm for massive classification tasks,” 2008 IEEE International Conference on Research, Innovation and Vision for the Future in Computing and Communication Technologies, Ho Chi Minh City, 2008, pp. 215-220.

[6] Q. Li, R. Salman and V. Kecman, “An intelligent system for accelerating parallel SVM classification problems on large datasets using GPU,” 2010 10th International Conference on Intelligent Systems Design and Applications, Cairo, 2010, pp. 1131-1135.

[7] Andreas Athanasopoulos, Anastasios Dimou, Vasileios Mezaris, Ioannis Kompatsiaris, “GPU acceleration for support vector machines”, In Procs. 12th Inter. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2011)

[8] Sopya K., Drozda P., Grecki P., “SVM with CUDA Accelerated Kernels for Big Sparse Problems”. In: Rutkowski L., Korytkowski M., Scherer R., Tadeusiewicz R., Zadeh L.A., Zurada J.M. (eds) Artificial Intelligence and Soft Computing. ICAISC 2012. Lecture Notes in Computer Science, vol 7267. Springer, Berlin, Heidelberg

[9] Yunmei Lu, Yun Zhu, Meng Han, Jing (Selena) He, and Yanqing Zhang. 2014. “A survey of GPU accelerated SVM.” In Proceedings of the 2014 ACM Southeast Regional Conference (ACM SE ’14). ACM, New York, NY, USA, Article 15, 7 pages.

[10] SUSY Data Set, URL: https://archive.ics.uci.edu/ml/datasets/SUSY

[11] Confusion Matrix, URL: https://en.wikipedia.org/wiki/Confusion_matrix