J. KSIAM Vol.22, No.1, 15–28, 2018 http://dx.doi.org/10.12941/jksiam.2018.22.015

ACCELERATION OF MACHINE LEARNING ALGORITHMS BY TCHEBYCHEV ITERATION TECHNIQUE

MIKHAIL P. LEVIN

INSTITUTE OF SYSTEM PROGRAMMING OF RUSSIAN ACADEMY OF SCIENCES, MOSCOW, RUSSIA
E-mail address: Mikhail_Levin@hotmail.com; mlevin@ispras.ru

ABSTRACT. Recently Machine Learning algorithms are widely used to process Big Data in various applications and a lot of these applications are executed in run time. Therefore the speed of Machine Learning algorithms is a critical issue in these applications. However the most of modern iteration Machine Learning algorithms use a successive iteration technique well-known in Numerical Linear Algebra. But this technique has a very low convergence, needs a lot of iterations to get solution of considering problems and therefore a lot of time for processing even on modern multi-core computers and clusters. Tchebychev iteration technique is well-known in Numerical Linear Algebra as an attractive candidate to decrease the number of iterations in Machine Learning iteration algorithms and also to decrease the running time of these algorithms those is very important especially in run time applications. In this paper we consider the usage of Tchebychev iterations for acceleration of well-known K-Means and SVM (Support Vector Machine) clustering algorithms in Machine Leaning. Some examples of usage of our approach on modern multi-core computers under Apache Spark framework will be considered and discussed.

1. INTRODUCTION

Now a day Machine Learning algorithms are widely uses in Data Mining for solution of various application problems related with Big Data processing. In this paper we restrict our consideration by two very popular Machine Learning (ML) algorithms, namely K-Means and SVM (Support Vector Machine). These algorithms are among of the 10 the most popular algorithms [1] in ML and they are widely used in various applications.

K-Means and SVM algorithms are widely used in Machine Vision, Drag Design, Genomic and Bio-informatics, in Medical Cybernetics, in Finance and some other applications. For example, K-Means is used

- for direct clustering of Big Data;
- in Computer Graphics for color quantization (reducing color palette of images to a fixed number of colors;
- for preliminary acceleration of Community Detection clustering methods;
- as a preliminary procedure for parallelization of SVM clustering method.

Received by the editors May 11 2017; Accepted December 18 2017; Published online February 15 2018.

In Computer Vision K-Means clustering technology is one of the basic technologies, because it is used in 6 of 8 branches of Computer Vision, namely in: Signal Processing and Compression, Data Mining, Machine Learning and Artificial Intelligence, Computer Graphics, Automatic Control and Robotics, Applied Mathematics. Other two branches of Computer Vision, namely Physics Imaging and Neurobiology also use this technology but not so often as previous six.

One of attractive properties of K-Means clustering algorithm is its simplicity. But the payment for simplicity is weak properties of this algorithm, those are

- a lot of iterations especially in case of very big data and correspondingly a very big CPU time for this data processing;
- convergence to a local minimum.

The first of above mentioned weak point is a principle obstacle in K-Means clustering, because a lot of applications especially in Computer Vision are running in run time. This means that the time of reaction in these applications should be very small. Therefore decreasing of this time is one of the critical issue in these applications and it is very important and urgent problem in the usage of K-Means clustering.

2. Background of K-Means Clustering

K-Means clustering algorithm was proposed by Steinhaus in 1956 and developed by Lloyd in 1957. It takes about KN operations for clustering, where K is a number of clusters and N is a number of points in data. In Lloyd version K-Means comprises of the following steps:

- It starts by setting initial centers of clusters (so called seeding);
- Then assigns each data point to the closest cluster be evaluating the distance between each data point to each cluster centers and allocate each point to the nearest cluster;
- Re-evaluates each cluster center for each cluster group;
- Evaluates maximal absolute value of difference between centers on current and previous iterations;
- Repeats last three steps 2,3,and 4 until each cluster has stable center and members in appropriate cluster;
- Evaluates a sum of square error to estimate quality of clustering.

Evaluation of centers for each cluster in Lloyd's version of algorithm is provided as follows. Let any cluster with index k on i-th iteration consists of K_i points with coordinate vectors

$$p_j = \{p_{j1}, \quad p_{j2}, \quad ..., \quad p_{jn}\}, \quad j = 1, 2, 3, ..., K_i.$$

Then the center of k cluster on i iteration is evaluated as follows

$$x_i = \frac{1}{K_i} \sum_{j=1}^{K_i} p_j \tag{2.1}$$

Iterations are doing until

$$\max_k ||x_i - x_{i-1}|| < \varepsilon \tag{2.2}$$

Here ε is an exactness (usually about 10^{-6} and $||.||$ is Euclidean norm in space R_n. The scheme of classic Lloyd's version of K-Means algorithm is shown on Fig. 1.

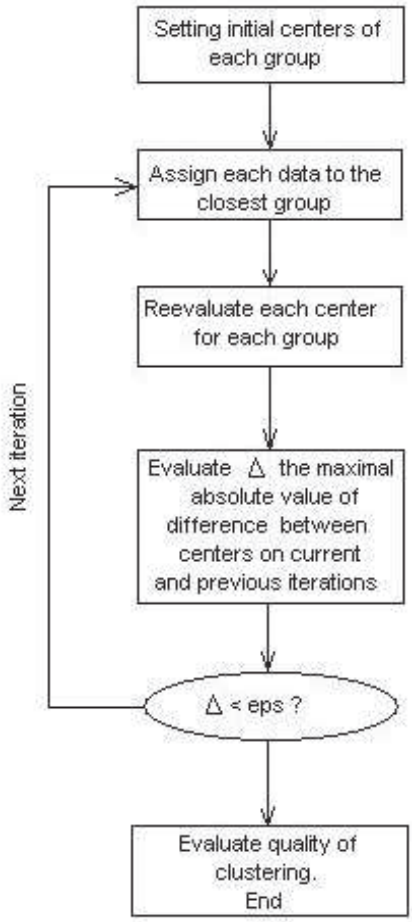

FIGURE 1. The scheme of Lloyd's version of K-Means clustering.

3. PROFILE-GUIDED TCHEBYCHEV ALGORITHM

To date it were a lot of attempts to speed-up K-Means clustering technology. Mainly these attempts were connected with changing of data structure, usage of another type of norm in distance evaluation instead of classical Euclidean norm. Here we should mention Dan Pelleg and Andrew Moore paper [2] in which kd-trees were used to accelerate nearest-neighbour search queries. This technique modifies steps 4 and 5 in Lloyd's algorithm by decreasing the number of operations on steps 4 and 5.

Another attempts were done in papers [3-7] and they are related with effect of seeding. In [3] a preclustering technique instead of random seeding was used to decrease the number of iterations and running time. Other algorithms of seeding were considered in [4-7]. All these modifications concern step 3 in Lloyd's algorithm.

Jenks in [8] suggested to use modified function in minimization problem and also suggested another seeding. He obtained about 90 percent speed-up of Lloyd's method, but only in some cases.

In 2004 Ya Guan, Ali Ghobani, and Nabil Belacel proposed K-Means+ modification [9]. It selects the number of clustering automatically basing on initial data analysis.

Unfortunately in all above cited approaches there were not any attempts to change iteration process and decrease by this manner the number of iterations. In all these approaches the successive iteration technique is used. Now we suggest to use the Tchebychev iteration technique instead successive iteration technique. In this case at first we represent the formula of clusters computation in evaluation form as follows

$$x_i - x_{i-1} \ = \ \frac{1}{K_i} \sum_{j=1}^{K_i} p_j \ - \ \frac{1}{K_{i-1}} \sum_{j=1}^{K_{i-1}} p_j \tag{3.1}$$

or as follows

$$x_i \ = \ x_{i-1} \ + \ \tau \, A(x_{i-1}, x_{i-2}) \tag{3.2}$$

Here

$$A(x_{i-1}, x_i) \ = \ \frac{1}{K_i} \sum_{j=1}^{K_i} p_j \ - \ \frac{1}{K_{i-1}} \sum_{j=1}^{K_{i-1}} p_j, \qquad \tau = 1 \tag{3.3}$$

Now let us consider four layers Tchebychev iteration method [10] for centers of clusters evaluation. We denote $\lambda_{\min}$ and $\lambda_{\max}$ minimal and maximal eigenvalues of operator A.

Also we denote so called "optimal" time step in the successive iteration method as follows

$$\tau_0 \ = \ \frac{2}{\lambda_{\min} \ + \ \lambda_{\max}}$$

and value

$$\varrho_0 \ = \ \frac{\lambda_{\max} \ - \ \lambda_{\min}}{\lambda_{\min} \ + \ \lambda_{\max}}$$

We suggest that the time step τ in iteration process is changing with respect to the number of iteration $(i + m)$ and evaluate it on each sequential 4 iterations by the following formulas

Let $i = 4q + m$, then

$$\tau_{(4q+m)} \ = \ \frac{\tau_0}{1 \ + \ \varrho_0 \ \cos[(2m \ - \ 1)\pi/8]} \, , \quad (m \ = \ 1, 2, 3, 4)$$

Here $\tau_{(4q+m)}$, $(m \ = \ 1, 2, 3, 4)$ are roots of Tchebychev polynomial of the 4th order. Thus we evaluate the centers of clusters by the following iteration formulas

$$x_i \ = \ x_{i-1} \ + \ \tau_{(4q+m)} \, A(x_{i-1}, x_{i-2}) \, , \quad i \ = \ (4q \ + \ m) \, , \quad (m \ = \ 1, 2, 3, 4) \tag{3.4}$$

Unfortunately in the formulas above we can not expressed operator A in the explicit analytical form and therefore we can not evaluate or estimate its maximal and minimal eigenvalues analytically or numerically. At the first glance this problem is seemed unresolvable in mathematical sense. To solve it we go out of mathematics and go into informatics. To solve our problem we formulate two important properties basing on which we try to solve our problem.

Property 1: *Operator A should be compressible to provide the convergence of iteration process.*

Property 2: T*he number of iterations in the considering iteration process should be at least minimal as possible.*

Basing **Property 1** and **Property 2** we suggest the following algorithm to evaluate the maximal and minimal eigenvalues of operator A.

Profile-Guided Tchebychev Algorithm:

- **STEP 1**. Let us choose any small data subset d of the small volume from the considering data set $D = p_1, p_2, p_3, ..., p_K$. Here K is a number of points in the considering data set D.
- **STEP 2**. The volume of subset d should be small to provide evaluations by Lloyd's algorithm in a short time. In practice it is possible to use a random choice selection of subset d from D data set.
- **STEP 3**. Providing evaluations clusters centers with iteration formulas (3.4) on subset d for any fixed $\lambda_{\max}$ and $\lambda_{\min}$, we consider the following unrestricted minimization problem for maximal and minimal eigenvalues choosing as follows:

$$Find \quad N_{iter}(\lambda_{\max}, \lambda_{\min}) \to \min$$

The solution of above mentioned problem should be obtained by well-known alternative descent method [11]. The example of appropriate MatLab code is available in [12].

The scheme of profile-guided Tchebychev version of K-Means clustering algorithm is shown on Fig. 2. On this scheme the new blocks in comparison with Lloyd's version of K-Means algorithm are shown in oval frames. The overhead of the proposed modification is very small in comparison with evaluation of centers of clusters computation. It should be omit, because the time steps can be tabulated in advance and it needs only to provide evaluations by formula (3.2). These additional evaluations contain only 1 add and 1 multiply operations.

4. Performance of Profile-Guided Tchebychev Algorithm

Now let us consider an example of usage 4 Layers Profile-Guided Tchebychev Algorithm (4LPGTch) for K-Means clustering on the real well-known data file kmeans_ train_ 3089 consisting of 3089 lines with 5 columns. An exactness of convergence in this example was taken

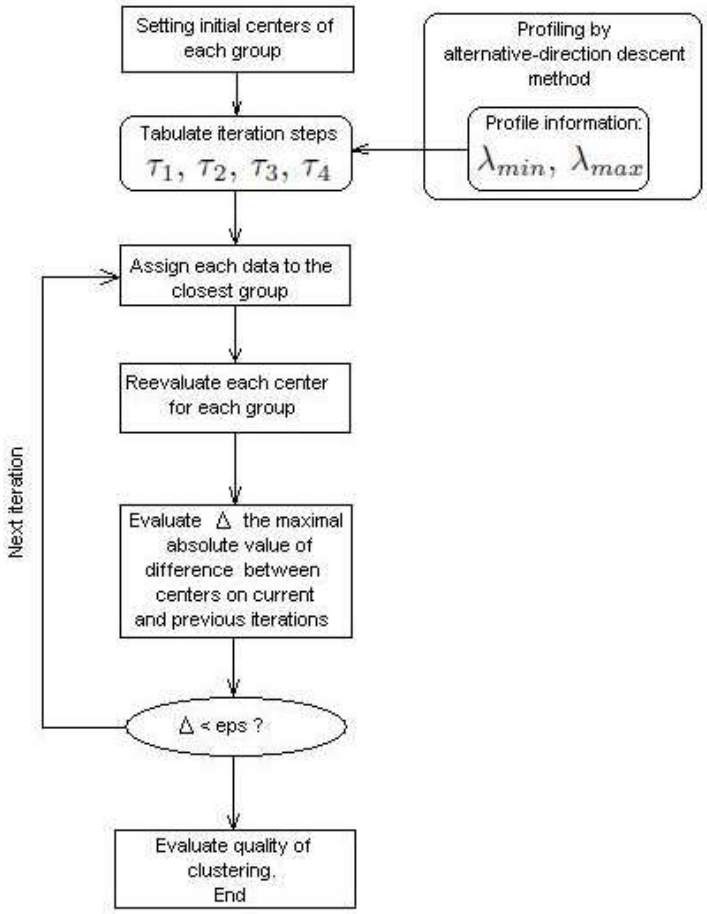

FIGURE 2. Profile-Guided Tchebychev 4 Layers K-Means Clustering.

equal to 10^{-7}. For comparison with our approach Successive Iteration Method (Lloyd's algorithm), 3 Layers B.T.Polyak's Iteration Method (3LPIM) [13], and Successive Over Relaxation Method (SORM) with fitting of changing relaxation coefficients were taken. The appropriate results are presented in the following Table 1.

Thus the usage of 4 Layers Profile-Guided Tchebychev Algorithm essentially decreases the number of iterations (up to $13/4 = 3.25$) times in comparison with successive iteration algorithm uses in Lloyd's version of K-Means clustering. In comparison with other iteration algorithms such as 3 Layers B.T.Polyak's Iteration Method and Successive Over Relaxation Method with fitting of changing relaxation coefficients, it has $11/4 = 2.75$ and $7/4 = 1.75$ advantage.

TABLE 1. Comparison of various algorithms for K-Means clustering

Algorithm	Number of iterations	Speed up
Lloyd's algorithm	13	0%(basement)
3LPIM	11	+15%
SORM	7	+46%
4LPGTch	4	+69%

Now let us consider one additional example of big volume data file whose was generated by repeating of 2000 times of kmeans_ train_ 3089 data. We provided computations by $Kmeans\|$ algorithm from Apache Spark ML Library. This algorithm is written on Scala programming language. Also we modified this software with 4, 6, and 8 Layers Profile-Guided Tchebychev ideas. We denote this version as $MLPGTch\_Kmeans\|$. Since $Kmeans\|$ and $MLPGTch\_Kmeans\|$ use random initial data in seeding of cluster centers, their run times are changing also randomly. Therefore for estimation of results we need to use some statistic estimations. We used the sets of 50 runs. Of course, this statistic is not perfect, but it gives us some raw statistic estimations of performance of considering algorithms. The appropriate results are presented in Table 2 below. The number of clusters in these examples was taken equal to 5 and the number of layers M was taken qual to 4, 6 and 8. The following notations are used avNI is an average number of iterations, avIT is an average time of iteration block run in second, avTT is an average total time run in seconds, and speedups su were evaluated as follows

$$su(avNI) = \frac{avNI(KMeans\|)}{avNI(MPGTcheb\_KMeans\|)},$$

$$su(avIT) = \frac{avIT(KMeans\|)}{avIT(MPGTcheb\_KMeans\|)},$$

$$su(avTT) = \frac{avTT(KMeans\|)}{avTT(MPGTcheb\_KMeans\|)}.$$

TABLE 2. Comparison of $KMeans\|$ and $MPGTch\_KMeans\|$ with $M\ 4,\ 6,\ 8$

Algorithm	avNI	avIT	avTT	su(avNI)	su(avIT)	su(avTT)
KMeans$\|\|_ML_Lib$	31	173"	197"	1.00	1.00	1.00
4PGTch$_KMeans\|\|$	23	130"	155"	1.36	1.34	1.27
6PGTch$_KMeans\|\|$	22	119"	141"	1.40	1.46	1.40
8PGTch$_KMeans\|\|$	22	134"	160"	1.40	1.29	1.23

This table shows that the best results are obtained with $6PGTch\_KMeans\|$ algorithm.

5. BACKGROUND OF SVM CLUSTERING

SVM (Support Vector Machine) algorithm was proposed by Vladimir Vapnik and Alexey Chervonenkis [1,14-15] in 1963. It widely uses in Machine Vision, Drag Design, Genomic and Bioinformatics, in Medical Cybernetic, in Finance for a direct clustering of big data.

In recent years SVM is very popular approach non-hierarchical clustering and binary classification. It can use a lot of different kernels and different type of measures to provide a best fitting of clustering data.

SVM has three important advantages:

- Firstly, it has a regularization parameter, which makes the user thinking about avoiding over-fitting;
- Secondary, it uses a kernel trick, so can build in expert knowledge about the problem via engineering the kernel;
- Thirdly, SVM is solved by usage a convex optimization problem, namely Quadratic Programming Problem (QPP) which avoids local minimization solutions and for which developed various efficient methods.

The weak points of SVM are

- a lot of iterations for big data analysis and therefore a big CPU time for clustering;
- SVM theory only really covers the determination of the parameters for a given value of regularization and kernel parameters;
- SVM moves the problem of over-fitting from optimizing the parameters to the model selection. Sadly the kernel models can be quite sensitive to over-fitting the model selection criterion.

To evaluate a separate hyper-plane parameters, SVM uses Quadratic Programming Problem (QPP). To date there are a few different methods for solution of QPP in SVM. One of the most promising approach was proposed by John C. Platt [16] from Microsoft Research. In this approach instead the full multidimensional QPP the sequence of 2 dimensional QPP is solved step-by-step in successive iteration process.

Thus one of the challenge problem in SVM clustering is an acceleration of convergence and decreasing c CPU time, especially in run-time applications, there the response time is a critical value, and in big data analysis.

In a linear case SVM problem consists in evaluation of a separate plane

$$u \equiv \overrightarrow{w} \cdot \overrightarrow{u} - b = 0 \tag{5.1}$$

subject to the following inequality

$$y_i(\overrightarrow{w} \cdot \overrightarrow{x_i}) - b \geq 0, \quad 1 \leq i \leq n \tag{5.2}$$

Here u is output of SVM, b is a threshold, $\overrightarrow{x_i}$ is a training example to the input $\overrightarrow{x}$, $\overrightarrow{y_i} \in \{-1, +1\}$ is a desired output and n is a dimension of problem.

The problem of searching the separate plane (5.1) subject to conditions (5.2) is reduced to minimization of $\|\overrightarrow{w}\|$ subject to conditions (5.2). According to Kurosh-Kuhn-Tucker theorem

(KKT) the above mentioned problem of Quadratic Optimization (QOP) is equivalent to the dual problem of the saddle point to Lagrange function evaluation as follows

$$L(\overrightarrow{w},\, b,\, \overrightarrow{\alpha}) \,\equiv\, \frac{1}{2}\|\overrightarrow{w}\|^{\,2} \,-\, \sum_{i=1}^{n} \overrightarrow{\alpha}_i y_i((\overrightarrow{w} \,\cdot\, \overrightarrow{x}_i) \,-\, b) \,\rightarrow\, \min_{(\overrightarrow{w},b)}\, \max_{\overrightarrow{\alpha}}. \tag{5.3}$$

Here $\overrightarrow{\alpha} \,=\, \{\alpha_i\}_{i=1}^{n}$ are Lagrange multipliers.

QOP (5.3) can be rewritten with respect to Lagrange multipliers as the following Quadratic Programming Problem (QPP)

$$-\,L(\overrightarrow{\alpha}) \,\equiv\, -\,\sum_{i=1}^{n} \overrightarrow{\alpha}_i \,+\, \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j (\overrightarrow{x_i}\cdot\overrightarrow{x_j}) \,\rightarrow\, \max_{\overrightarrow{\alpha}}, \tag{5.4}$$

$$0 \,\leq\, \overrightarrow{\alpha_i} \,\leq\, C, \;\; 1 \,\leq\, i \,\leq\, n \quad , \tag{5.5}$$

$$\sum_{i=1}^{n} \alpha_i y_i \,=\, 0 \quad . \tag{5.6}$$

Training of SVM consists in evaluation of $\overrightarrow{\alpha}$ from solution of QPP (5.4-5.6) and evaluation of $\overrightarrow{w}$ and b as follows

$$\overrightarrow{w} \,=\, \sum_{i=1}^{n} \alpha_i \, y_i \, \overrightarrow{x_i} \,, \tag{5.7}$$

$$b \,=\, \overrightarrow{w} \,\cdot\, \overrightarrow{x_i} \,-\, y_i \,, \quad \alpha_i \,>\, 0 \;. \tag{5.8}$$

SMO algorithm consists in iterative solution step-by-step with respect to any pair of Lagrange multipliers α_l, α_m. Each QP sub-problem with respect to α_l, α_m is solved analytically. Iterations are terminated when all Karush-Kuhn-Tucker (KKT) optimality conditions has been valid

$$\text{if} \;\; \alpha_i \,=\, 0, \;\; \text{then} \;\; y_i u_i \,\geq\, 1 \,, \tag{5.9}$$

$$\text{if} \;\; 0 \leq\, \alpha_i \,<\, C, \;\; \text{then} \;\; y_i u_i \,=\, 1 \,, \tag{5.10}$$

$$\text{if} \;\; \alpha_i \,=\, C, \;\; \text{then} \;\; y_i u_i \,\leq\, 1 \,. \tag{5.11}$$

The SVM-SMO scheme of principle is shown on Fig. 3.

6. Profile-Guided Tchebychev modification of SVM-SMO Clustering

Now let us consider a modification of SVM-SMO algorithm using ideas of Profile-Guided Tchebychev approach applied before to K-Means algorithm. At first we present formulas for parameters of separation plane computation in evaluation form

$$\overrightarrow{w}^{(k)} \,-\, \overrightarrow{w}^{(k-1)} \,=\, \sum_{i=1}^{n} \alpha_i^{(k)} \, y_i \, \overrightarrow{x_i} \,-\, \overrightarrow{w}^{(k-1)} \,, \tag{6.1}$$

$$b^{(k)} \,-\, b^{(k-1)} \,=\, \overrightarrow{w}^{(k)} \,\cdot\, \overrightarrow{x_i} \,-\, y_i \,-\, b^{(k-1)} \,, \quad \alpha_i \,>\, 0 \;. \tag{6.2}$$

Formulas (6.1-6.2) can be presented in the following vector form

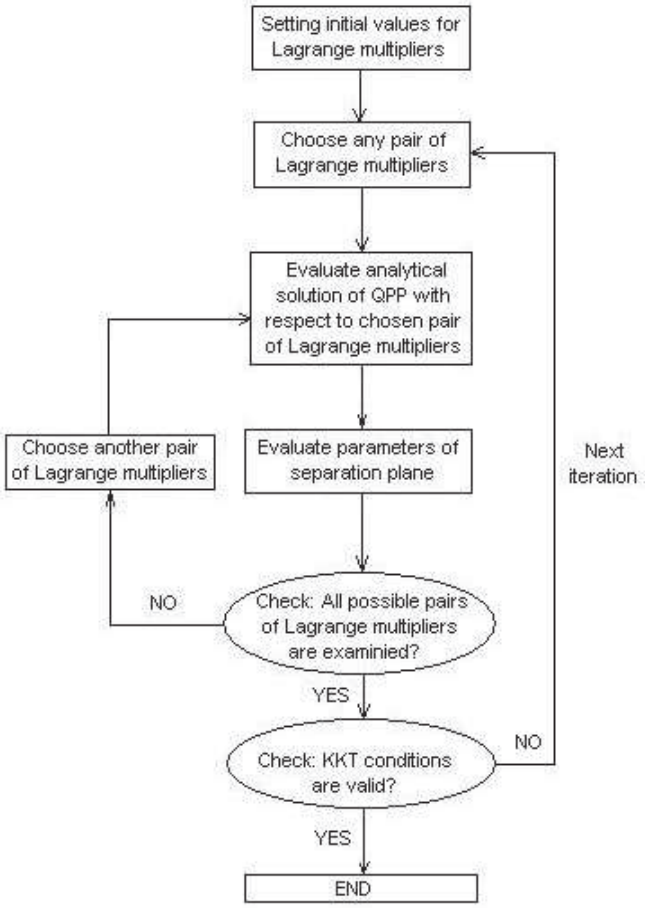

FIGURE 3. SVM-SMO scheme of principle.

$$\overrightarrow{Z}^{(k)} \;=\; \overrightarrow{Z}^{(k-1)} \;+\; \tau \overrightarrow{\mathbf{A}}(\alpha_i^{(k)},\; y_i,\; \overrightarrow{x_i},\; \overrightarrow{z}^{(k-1)})\,, \tag{6.3}$$

where $\overrightarrow{\mathbf{A}} \;=$

$$\left\{ \begin{array}{c} \sum\limits_{i=1}^{n} \alpha_i^{(k)}\; y_i\; \overrightarrow{x_i} \;-\; \overrightarrow{w}^{(k-1)} \\ \overrightarrow{w}^{(k)} \;\cdot\; \overrightarrow{x_i} \;-\; y_i \;-\; b^{(k-1)} \end{array} \right\}$$

is a vector operator, $\tau = 1$, k is a number of iteration, and $\overrightarrow{Z} \;= \{\overrightarrow{w},\; b\}$ is a searching vector.

Formula (6.3) represents a successful iteration technique for evaluation of parameters of separation plane. Now we will consider application of Multi-Layers Tchebychev iteration technique for evaluation of parameters of separation plane in SVM-SMO algorithm.

Let τ is not constant and is changing in iteration process with respect to the number of iterations and is evaluated on each M iterations as follows

$$\tau_{(k+m)} \ = \ \frac{\tau_0}{1 \ + \ \varrho_0 \, cos[(2m \ - \ 1)\pi/(2M)]} \ , \quad (m \ = \ 1, 2, 3, ..., M) \ ,$$

where

$$\tau_0 = \frac{2}{\lambda_{\min} + \lambda_{\max}}, \quad \varrho_0 = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\min} + \lambda_{\max}},$$

$\lambda_{\min}$ and $\lambda_{\max}$ are minimal and maximal eigenvalues of operator $\overrightarrow{\mathbf{A}}$.

Now let us consider Multi-Layers Profile-Guided Tchebychev (MLPGTch) algorithm for SVM-SMO clustering. This algorithm is quite similar to 4LPGTch algorithm considered before for K-Means clustering, but it has additional parameter the number of layers M.

Multi-Layers Profile-Guided Tchebychev Algorithm:

- 1.Let us choose any small data subset d of the small volume from the considering data set $D = p_1, p_2, p_3, ..., p_K$. Here K is a number of points in the considering data set D.
- 2.The volume of subset d should be small to provide fast evaluations by SVM-SMO algorithm in a short time. In practice it is possible to use a random choice selection of subset d from D data set.
- 3. Providing evaluations of separation plane by iteration formulas (6.3) on subset d for any fixed number of layers M, eigenvalues $\lambda_{\max}$ and $\lambda_{\min}$, we consider the following unrestricted minimization problem for maximal and minimal eigenvalues, and the number of layers choosing as follows:

  $$\text{Find } N_{iter}(M, \lambda_{\max}, \lambda_{\min}) \to \min$$

  The solution of above mentioned problem also as for K-Means algorithm can be obtained by well-known alternative descent method.

The scheme of multi-layers profile-guided Tchebychev version of SVM-SMO clustering algorithm is shown on Fig. 4. On this figure the new blocks in comparison with SVM-SMO algorithm are shown by rectangles with rounded corners.

In our numerical profiling we restricted consideration by case $M = \{4, 6, 8\}$. Solving minimization problem of test data set of small volume we obtained the following solution: $M = 6$, $\lambda_{\max} = 1.0010$, $\lambda_{\min} = 0.7955$. As our experience shows it is possible to use this solution in MLPGTch clustering for other data sets including data sets of huge volume. Perhaps this is related with weak dependence of operator $\overrightarrow{\mathbf{A}}$ with data sets. But this is only our hypothesis confirmed by our numerical experiments. Below we consider some numerical results obtained with MLPGTch algorithm.

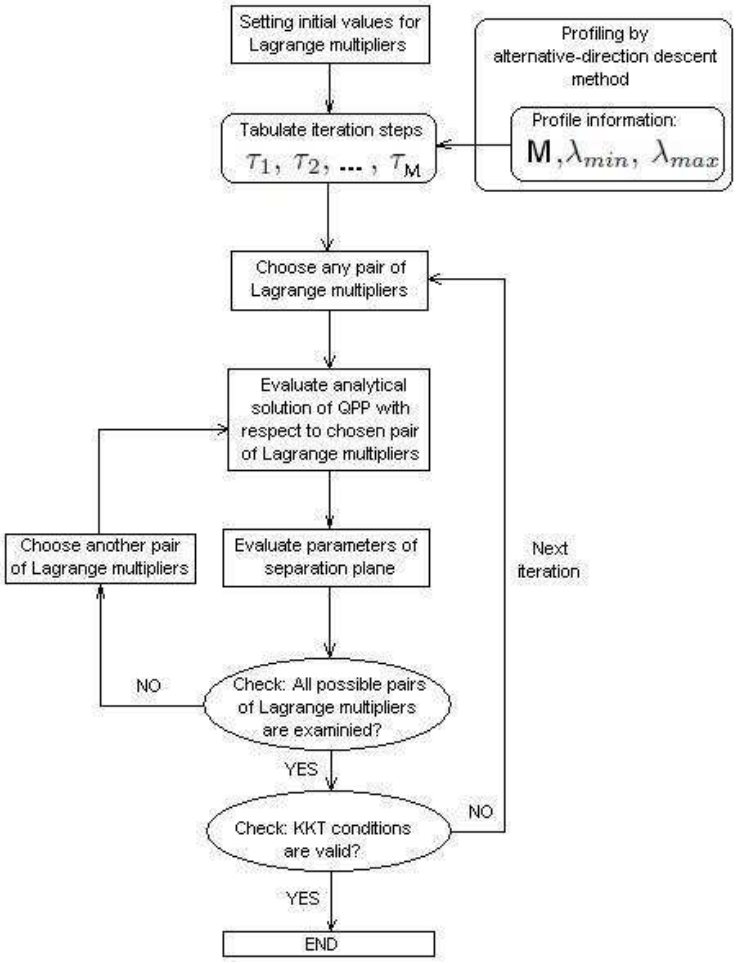

FIGURE 4. MLPGTch SVM-SMO scheme of principle.

7. PERFORMANCE OF MULTI-LAYERS PROFILE-GUIDED TCHEBYCHEV SVM-SMO ALGORITHM

Now let us consider some examples of usage Multi-Layers Profile-Guided Tchebychev SVM-SMO Algorithm for clustering on some examples. At first we consider two simple 2D examples for which it is possible to construct the separation line by using the symmetry properties of data. The appropriate data are accumulated in the following Table. 3. The exactness of convergence in all examples was taken equal to 10^{-4}. For comparison with our approach original SVM-SMO algorithm was taken. The appropriate results are presented in the following Table 4.

Table 3. Data for Example 1 and Example 2

Data	Example 1	Example 2
x matrix	$\begin{pmatrix} -3.0 & 2.0 \\ -2.0 & 2.0 \\ -2.0 & 3.0 \\ 2.0 & -3.0 \\ 3.0 & -2.0 \\ 2.0 & -2.0 \end{pmatrix}$	$\begin{pmatrix} 2.0 & 2.0 \\ 2.0 & 1.0 \\ 1.0 & 2.0 \\ -2.0 & -1.0 \\ -2.0 & -2.0 \\ -1.0 & -2.0 \end{pmatrix}$
y vector	$\begin{Bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \end{Bmatrix}$	$\begin{Bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \end{Bmatrix}$

Here Niter(Algorithm) is a number of iterations with usage of the appropriate algorithm and Speed_up=Niter(SVM-SMO)/Niter(MLPGTch)

Table 4. Comparison of algorithms for SVM-SMO clustering

Algorithm	Number of iterations	Speed_up
Example 1: Original SVM-SMO	162	1.00
Example 1: MLPGTch M=6	99	1.64
Example 2: Original SVM-SMO	709	1.00
Example 2: MLPGTch M=6	54	13.13

Thus we can see that usage of Multi-Layers Profile-Guided Tchebychev Algorithm essentially decreases the number of iterations (up to 13.13) times in comparison with successive iteration algorithm used in original SVM-SMO.

8. CONCLUSIONS

The presented results show that Multi-Layers Profiled-Guided Tchebychev technique can be effective in acceleration of iteration algorithms in Machine Leaning.

REFERENCES

[1] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. MacLachlan Angus Ng, Bing Liu, PhilipS. Yu, Zhi-Hau Zhou, Michael Steinbach, David J. Hand, and Dan Steinberd, *Top 10 Algorithms in Data Mining*, - In: Knowledge Information Systems, 2008, 14, pp. 1–37.

[2] Dan Pelleg and Andrew Moore, *Accelerating Exact k-means Algorithms with Geometric Reasoning*, - In: KDD '99 The Fifth International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA - August 15-18, 1999; ACM New York, NY, USA, 1999, pp. 277–281.

[3] Phillip Eliasson and Niklas Rosen, *Efficient K-means clustering and the importance of seeding*, - In: KTH Royal Institute of Technology, School of Computer Science and Communication Reports, (http://www.csc.kth.se/utbilding/kth/kurser/DD143X/kand13/Group4Per/report/ 40-eliasson-rosen.pdf)

[4] Sugato Basu, Arindam Banerjee, and Raymond Mooney, *Semi-supervised Clustering by Seeding*, - In: Proceedings of the 19th International Conference on Machine Learning (ICML-2002), pp. 19–26, Sydney, Australia, July 2002.

[5] David Arthur, and Sergei Vassilvitskii, *k-means++: The advantages of Careful Seeding*, - In: Stanford InfoLab Publications, 2006.

[6] K. Karteeka Pavan, Allam Appa Rao, A. V. Dattatreya Rao, and G. R. Sridhar, *Robust seed selection algorithm for k-means type algorithms – Optimal centroids using high density object*, - In: International Journal of Computer Science & Information Technology (IJCSIT), vol.3, No 5, Oct 2011, pp. 148–163.

[7] Foud Khan, *An inial seed selection algorithm for k-means clustering of georeferenced data to improve replicability of cluster assignments for mapping application*, - In: Appied Soft Computing, vol. 12, issue 11, November 2012, pp. 3698–3700.

[8] G. F. Jenks, *The data model concept in statistical mapping*, - In: International Yearbook of Cartography 7 (1967), pp. 186–190.

[9] Yu Guan, Ali A. Ghorbani, and Nabil Bekacel, *K-means+: An Autonomous Clusterin Algorithm*, - In: (http://neuro.bstu.by/ai/To-dom/My_research/Papers-0/For-research/D-mining/Iris-kmean/kmeans-plus.pdf), 2004.

[10] A.A.Samarskii, *Introduction into Numerical Methods*, Moscow, Science Publisher, 1997, pp. 234.

[11] Stephen J. Wright, *Coordinate descent algorithms*, - In: Mathematical Programming 151 (1): pp. 3-34, 2015. arXiv1502.04759.

[12] http://www.cyberforum.ru/matlab/tread1330639.html

[13] B.T.Polyak, *Some methods of speeding up the convergence of iteration methods*, - In: USSR Computational Mathematics and Mathematical Physics, vol.4, issue 5, 1964, pp. 1–17.

[14] V.N.Vapnik, and A.Ya.Chervonenkis, *Theory of pattern recognition: Statistical problems of learning*, Moscow, Science Publisher, 1974, pp. 416.

[15] W.N.Wapnik, A.J.Tscherwonenkis, *Theorie der Zeichenerkennung*, Berlin, Akademie-Verlag, 1979, pp. 343.

[16] John C. Platt, *Using Analytic QP and Sparseness to Speed Training of Support Vector Machines*, - In: Advances in Neural Information Processing Systems 11, 1999, pp. 557–563.