A Multi-parameter Updating Fourier Online Gradient Descent Algorithm for Large-scale Nonlinear Classification

Yingying Chen

March 17, 2022

Abstract

Large scale nonlinear classification is a challenging task in the field of support vector machine. Online random Fourier feature map algorithms are very important methods for dealing with large scale nonlinear classification problems. The main shortcomings of these methods are as follows: (1) Since only the hyperplane vector is updated during learning while the random directions are fixed, there is no guarantee that these online methods can adapt to the change of data distribution when the data is coming one by one. (2) The dimension of the random direction is often higher for obtaining better classification accuracy, which results in longer test time. In order to overcome these shortcomings, a multi-parameter updating Fourier online gradient descent algorithm (MPU-FOGD) is proposed for large-scale nonlinear classification problems based on a novel random feature map. In the proposed method, the suggested random feature map has lower dimension while the multi-parameter updating strategy can guarantee the learning model can better adapt to the change of data distribution when the data is coming one by one. Theoretically, it is proved that compared with the existing random Fourier feature maps, the proposed random feature map can give a tighter error bound. Empirical studies on several benchmark data sets demonstrate that compared with the state-of-the-art online random Fourier feature map methods, the proposed MPU-FOGD can obtain better test accuracy.

1 Introduction

Support vector machines (SVM) are powerful tools for data classification in machine learning and have been widely applied into the fields of pattern recognition [ZOW16, CYXY16, HST, DNSV19, YDL19, NGB17], image processing [YSZ+18, GJP18, ZWZ+17], computer vision [KLD+19, ZSW+19], data mining [WSH+18, XZL+18], and so on. Generally speaking, the computational complexity of SVM is $O(n^3)$, which leads to that training SVM becomes a challenging task for large scale classification problems.

At present, the existing methods for large scale classification problems can be categorized into two classes: offline learning and online learning. In the case of offline learning, on the one hand, the researchers have proposed Lagrangian support vector machine algorithm [MM01], stochastic gradient descent algorithms [Zha04, SSS07], modified finite Newton algorithm [KD05], cutting-plane method [Joa06], Bundle method [LS08, TVSL10], coordinate descent method [CHL08], and dual coordinate descent method [HCL+08] to deal with large scale linear classification problems. On the other hand, the researchers have presented SMO algorithm [KSBM01], low-rank kernel representation method [SS00, FS01], reduced support vector machine [LM01], core vector machine [TKC05], and localized support vector machine [ZBMM06, CTJ09, CGLL10] to deal with large scale nonlinear classification problems. In machine learning, research on algorithms for large scale linear classification problems has been mature. Inspired by this fact, a finite-dimensional mapping based offline algorithm has been proposed to deal with large scale nonlinear classification problems [RR08]. Specifically, for the positive definite shift-invariant kernel, it first makes use of random Fourier features to map the input data to a randomized low-dimensional feature space, and then uses linear algorithms to deal with large scale nonlinear classification problems.

In the previous studies, the idea of converting a batch optimization problem into an online task was suggested in [LL00]. Inspired by [LL00] and [Ros58], an online learning framework [CDK+06] was suggested based on the passive-aggressive strategy to deal with linear classification, linear regression,
and uniclass prediction problems. The main difference between [Ros58] and [CDK+06] is that except for classification and regression, [CDK+06] can deal with uniclass prediction problem. For large-scale nonlinear classification problems, [WZH+13] extended [RR08] to online learning and proposed Fourier online gradient descent algorithm (FOGD), which often needs large number of random features $D$. [LWZ14] answered the question "How many random Fourier features are needed to obtain the better results in the online kernel learning setting?". A local online learning algorithm was proposed [ZZH+16] based on the observation that although the data sets are not globally linear separable, they may still be locally linear separable. Using the reparameterized random feature, a large-scale online kernel learning algorithm was proposed [NLBP17]. [JP18] suggested an online learning algorithm via combining the passive-aggressive strategy and max-out function (PAMO). Using the Nyström approximation technology, an online gradient descent (NOGD) algorithm was designed [LHW+16]. Unfortunately, it is not easy to approximate the whole kernel matrix using a submatrix. The smaller submatrix will degrade test accuracy while the larger submatrix will lead to higher computational complexity. For the random Fourier feature mapping based online learning, generally speaking, a larger $D$ leads to a more precise approximation. However, a larger $D$ leads to a higher computational complexity. To reduce the impact number of random features, recently, a dual space gradient descent algorithm (DualSGD) was proposed [LNNP16].

The main shortcomings of random Fourier feature mapping based online learning methods are as follows: (1) Since only the hyperplane vector is updated during learning while the random directions are fixed, there is no guarantee that these online methods can adapt to the change of data distribution when the data is coming one by one. (2) The dimension of the random direction is often higher for obtaining better classification accuracy, which results in longer test time. In order to overcome these shortcomings, in this study, a multi-parameter updating Fourier online gradient descent algorithm (MPU-FOGD) is proposed for large-scale nonlinear classification problems based on a novel random feature map. In the proposed method, the suggested random feature map has lower dimension while the multi-parameter updating strategy can guarantee the learning model can better adapt to the change of data distribution when the data is coming one by one. Theoretically, we prove that the proposed random feature map can give a tighter error bound. Empirical studies on several benchmark data sets show that compared with the state-of-the-art online random Fourier feature map methods, the proposed MPU-FOGD can obtain better test accuracy.

The rest of this paper is organized as follows. Section 2 describes the proposed feature map in details and presents theoretical guarantees. Section 3 gives experiment results and analysis. Section 4 encloses our paper with future work.

## 2 MPU-FOGD

### 2.1 Randomized Fourier Feature Map Algorithm

Let $m_t = \{x_t, l_t\}, t = 1, 2, \ldots, n$ be the training examples, where $n$ is the number of examples, $x_t \in \mathbb{R}^d$ is the input data, $l_t \in \{-1,+1\}$ and $l_t \in \{1, 2, \ldots, m\}$ are the corresponding label for binary classification and multi-class classification, respectively. Let $H$ be a Reproducing Kernel Hilbert Space (RKHS) endowed with a kernel function $k(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$.

For binary classification problems, the standard model of SVM is as follows:

\[
\min_{w,b} \frac{\lambda}{2}||w||_H^2 + \sum_{t=1}^{n} l(t; l_b) = \max(0, 1 - lb \cdot f(x_t)),
\]

where $\lambda > 0$, $f(x_t) = w^T \phi(x_t) + d$, $l (t; l_b) = max(0, 1 - lb \cdot f(x_t))$.

The computational complexity of this model is usually $O(n^3)$. For large scale nonlinear classification problems, it undoubtedly faces computational difficulty. Considering that research on SVM algorithms for large scale linear classification problems has been mature in machine learning, a randomized Fourier feature map $(\phi(x))^T = (\cos(u_1^T x), \cos(u_2^T x), \ldots, \cos(u_D^T x)), \sin(u_1^T x), \sin(u_2^T x), \ldots, \sin(u_D^T x))$ is suggested in [RR08] to breakthrough this bottleneck, where $D = O(\epsilon^2 \log(D/\epsilon))$, $\epsilon$ is the accuracy of approximation, the random vectors $u_i$ for $\forall i \in \{1, 2, \ldots, D\}$ are drawn from

\[
p(u) = \frac{1}{2\pi^d} \int e^{-iux^T} d(\Delta x).
\]
Obviously, using the trick in [RR08], the original data are mapped into a finite low-dimensional feature space and linear learning algorithms can be used to deal with large scale nonlinear classification problems. This is a very important contribution for large scale nonlinear classification problems in the field of machine learning.

2.2 Proposed Random Fourier Feature Map

Let $z(x, u)^T = (\cos(u^T x), \sin(u^T x))$, where $z(x)^T z(y) = \frac{1}{D} \sum_{j=1}^{D} z(x, u_j) z(y, u_j)$. According to Bochner’s theorem [GJP18], we know that $z(x)^T z(y)$ is an unbiased estimate of the positive definite shift-invariant kernel $k(x, y)$ [RR08]. From Claim 1 in [RR08], we know that the discrepancy between $k(x, y)$ and $z(x)^T z(y)$ can be narrowed down by varying $D$. Generally speaking, a larger $D$ will lead to a more precise approximation, but it will generate a higher computational complexity. Let $z_{old}(x)^T z_{old}(y) = \frac{1}{D} \sum_{j=1}^{D} z_{old}(x, u_j, b_j) z_{old}(y, u_j, b_j)$, where $z_{old}(x, u, b) = \sqrt{2} \cos(u^T x + b)$, $b$ is drawn uniformly from $[0, 2\pi]$. From [RR08], we know that $E(z_{old}(x)^T z_{old}(y)) = k(x, y)$, seeing Appendix A for details. Let $z_{new}(x)^T z_{new}(y) = \frac{1}{D} \sum_{j=1}^{D} z_{new}(x, u_j, b_j) z_{new}(y, u_j, b_j)$, where $z_{new}(x, u, b) = \sqrt{\frac{D}{2}} \cos(u^T x + b)$. In order to show that $z_{new}(x)^T z_{new}(y)$ is also an unbiased estimate of $k(x, y)$, we give the following Lemmas.

Based on the above analysis, $(\varphi(x))^T = (\sqrt{\frac{D}{2}} \cos(u_1^T x + b_1), \sqrt{\frac{D}{2}} \cos(u_2^T x + b_2), \cdots, \sqrt{\frac{D}{2}} \cos(u_D^T x + b_D))$ is adapted in this paper. Theoretically, this map will be able to obtain the better accuracy.

2.3 Proposed Algorithm

In the existing work, what non-stationary streaming learning environments [GMCR04, FBdCÁRJ+14, PV16] emphasize more is whether the change of posterior probability causes the change of its classification decision boundary. Learning from streaming data with concept drift, they usually have a detector to detect whether the distribution of data has changed. No matter what kind of detector they use, if the error rate changes significantly, then concept drift occurs, otherwise the current single classifier is used to predict examples and update model. On the other hand, Fourier feature mapping is obviously data independent. In the previous study, only the hyperplane vector $w$ is updated during learning while the random directions $u$ and $b$ are fixed. Obviously, there is no guarantee that the random Fourier-feature-mapping-based large scale online learning approaches could adapt to the change of data distribution. So, in this study, inspired by streaming learning, we iteratively update random directions $u$ and $b$ which can change the data distribution in the feature space and make the data better fit in with the current space.

MPU-FOGD includes two algorithms, i.e. MPU-FOGDU and MPU-FOGDUB. In comparison with the algorithm of MPU-FOGDUB, MPU-FOGDU only updates Fourier component $u$. Following the framework of large scale online kernel learning [WZH+13], when the loss function $l(l(b_t, w_t^T \varphi(x_t))) > 0$ for the $t$-th incoming sample, the update formulas of $u_{t+1}$, $b_{t+1}$ and $w_{t+1}$ in the proposed MPU-FOGD are as follows:

$$w_{t+1} = w_t - \eta_1 \nabla_{w_t} l(l(b_t, w_t^T \varphi(x_t))),$$

$$u_{t+1} = u_t - \eta_2 \nabla_{u_t} l(l(b_t, w_t^T \varphi(x_t))),$$

$$b_{t+1} = b_t - \eta_3 \nabla_{b_t} l(l(b_t, w_t^T \varphi(x_t))),$$

where

$$\nabla_{w_t} l(l(b_t, w_t^T \varphi(x_t))) = -l(b_t \varphi(x_t)),$$

$$\nabla_{u_t} l(l(b_t, w_t^T \varphi(x_t))) = -\sqrt{2} D l(b_t w_t^T \star (\sin(u_{t,1}^T x_t + b_{t,1}), \sin(u_{t,2}^T x_t + b_{t,2}), \cdots, \sin(u_{t,D}^T x_t + b_{t,D})) * (x_t, x_t, \cdots, x_t),$$

$$\nabla_{b_t} l(l(b_t, w_t^T \varphi(x_t))) = \sqrt{2} D l(b_t w_t^T \star (\sin(u_{t,1}^T x_t + b_{t,1}), \sin(u_{t,2}^T x_t + b_{t,2}), \cdots, \sin(u_{t,D}^T x_t + b_{t,D}))),$$

where $\star$ denotes dot product.
Based on the analysis mentioned above, the detailed steps of the proposed MPU-FOGD algorithm for binary classification are given in Algorithm 1.

Algorithm 1: the MPU-FOGD algorithm

**Data:** $D, \eta, \sigma$

**Output:** model: $w_{t+1} \in R^D, u_{t+1} \in R^{d \times D}, b_{t+1} \in R^D$

1. initial $w_0 = 0$;
2. initial $u_{0,i}$ for $\forall i \in \{1, 2, \cdots, D\}$ are drawn from $Dp(u)$;
3. initial $b_{0,i}$ for $\forall i \in \{1, 2, \cdots, D\}$ are drawn from $[0, 2\pi]$;
4. forall $t$ of $[T]$ do
5. receive instance: $x_t \in R^d$;
6. compute instance representation:
7. $(\varphi(x_t))^T = (\sqrt{\frac{2}{D}} \cos(w_{1,i}x_t + b_{1,i}), \sqrt{\frac{2}{D}} \cos(w_{2,i}x_t + b_{2,i}), \cdots, \sqrt{\frac{2}{D}} \cos(w_{D,i}x_t + b_{D,i}))$;
8. predict $\hat{b}_t = \text{sign}(w_t^T \varphi(x_t))$;
9. receive correct label: $lb_t \in \{-1, 1\}$;
10. compute hinge loss:
11. $l(\hat{b}_t, w_t^T \varphi(x_t)) = \max\{0, 1 - \hat{b}_t(w_t^T \varphi(x_t))\}$;
12. if $l(\hat{b}_t, w_t^T \varphi(x_t)) > 0$ then
13. $w_{t+1} = w_t - \eta \nabla_{w_t} l(\hat{b}_t, w_t^T \varphi(x_t))$;
14. $u_{t+1} = u_t - \eta \nabla_{u_t} l(\hat{b}_t, w_t^T \varphi(x_t))$;
15. $b_{t+1} = b_t - \eta \nabla_{b_t} l(\hat{b}_t, w_t^T \varphi(x_t))$;
16. end
end

In the multi-class problem setting, the algorithm predicts a sequence of scores for the $m$ classes:

$$(f_{t,1}(x_t), f_{t,2}(x_t), \cdots, f_{t,m}(x_t)).$$

The hinge loss in [LHW16] function as follows:

$$l(\gamma_t) = \max(0, 1 - \gamma_t),$$

where $\gamma_t = f_{t,l_b}(x_t) - f_{t,s_t}(x_t), s_t = \text{argmax}_{r \in Y, r \neq l_b} f_{t,r}(x_t)$.

3 Experiments

In this section, we empirically validate performance of the proposed algorithm over various benchmark data sets which can be freely downloaded from the LIBSVM\(^1\) and UCI\(^2\) website. Table 1 shows the details of 12 publicly available data sets. Mnist600k is obtained by randomly extracting 600,000 samples from the mnist8m\(^3\) data set after dimension reduction and normalization. All algorithms are implemented in Python 3.7.3, on a windows machine with AMD Ryzen 5 2600 Six-Core Processor\(\oplus\) 3.4GHZ. The codes of the proposed algorithm are written by using the online learning algorithm library [HWZ14].

We compare the proposed MPU-FOGD algorithm with online gradient descent algorithm (OGD) [Zin03], online passive aggressive algorithm (PA) [CDK06], online logistic regression (LR) [BDPDP96], Fourier online gradient descent algorithm (FOGD) [WZH13], Nyström online gradient descent algorithm (NOGD) [LHW16], reparameterized random feature algorithm (RRF) [NLBP17], hybrid the passive-aggressive strategy and the max-out function algorithm (PAMO) [JP18], and approximation vector machine (AVM) [LNNP17]. Except that OGD, PA, and LR are linear algorithms, the others are non-linear. PAMO is a binary classification algorithm, so we don’t do multi-class experiments for it in this study.

---

\(^1\)https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

\(^2\)https://archive.ics.uci.edu/ml/datasets.php
| Dataset     | Instances | Features | Classes |
|------------|------------|----------|---------|
| KDDCUP08   | 102,294    | 117      | 2       |
| ijcnn1     | 141,691    | 22       | 2       |
| codrma     | 271,617    | 8        | 2       |
| w7a        | 24,692     | 300      | 2       |
| skin       | 245,057    | 4        | 2       |
| covtype    | 581,012    | 54       | 2       |
| poker      | 1,000,000  | 10       | 10      |
| mnist600k  | 600,000    | 50       | 10      |
| forest     | 581,012    | 54       | 7       |
| acoustic   | 98,528     | 50       | 3       |
| aloi       | 108,000    | 128      | 1000    |
| combined   | 98,528     | 100      | 3       |

Table 1: An example table.

| Algorithm  | skin       | Test(s) | TestAcc(%) | kddcup08  | Test(s) | TestAcc(%) |
|------------|------------|---------|------------|-----------|---------|------------|
| LR         | 2.98       | 0.44    | 90.233     | 1.22      | 0.18    | 100.000    |
| OGD        | 2.51       | 0.38    | 90.068     | 0.80      | 0.16    | 100.000    |
| PA         | 5.15       | 0.38    | 81.630     | 1.96      | 0.16    | 100.000    |
| FOGD       | 15.44      | 3.52    | 99.942     | 2.39      | 0.49    | 100.000    |
| NOGD       | 75.12      | 7.06    | 99.047     | 28.29     | 2.79    | 99.452     |
| RRF        | 9.98       | 2.38    | 99.949     | 2.40      | 0.48    | 100.000    |
| PAMO       | 99.58      | 50.78   | 99.829     | 5.05      | 1.29    | 100.000    |
| AVM        | 10.12      | 1.32    | 99.616     | 191.03    | 1.14    | 99.717     |
| MPU-FOGDU  | 5.83       | 1.03    | 99.951     | 2.59      | 0.47    | 100.000    |
| MPU-FOGDUB | 6.26       | 1.11    | 99.954     | 2.69      | 0.47    | 100.000    |

| Algorithm  | ijcnn1     | Test(s) | TestAcc(%) | w7a       | Test(s) | TestAcc(%) |
|------------|------------|---------|------------|-----------|---------|------------|
| LR         | 1.72       | 0.26    | 67.672     | 0.32      | 0.05    | 95.556     |
| OGD        | 1.72       | 0.22    | 71.670     | 0.26      | 0.05    | 96.457     |
| PA         | 3.34       | 0.22    | 70.022     | 0.61      | 0.05    | 95.681     |
| FOGD       | 11.16      | 2.59    | 98.014     | 8.34      | 2.04    | 98.477     |
| NOGD       | 15.46      | 1.78    | 90.572     | 52.33     | 3.92    | 97.062     |
| RRF        | 18.83      | 3.63    | 97.780     | 208.44    | 37.91   | 98.494     |
| PAMO       | 191.55     | 43.60   | 98.687     | 28.62     | 6.25    | 98.664     |
| AVM        | 430.71     | 1.43    | 94.062     | 35.48     | 4.90    | 98.360     |
| MPU-FOGDU  | 21.29      | 3.01    | 98.885     | 25.23     | 0.55    | 98.388     |
| MPU-FOGDUB | 23.35      | 3.00    | 98.885     | 24.77     | 0.54    | 98.396     |

| Algorithm  | codrma     | Test(s) | TestAcc(%) | covtype   | Test(s) | TestAcc(%) |
|------------|------------|---------|------------|-----------|---------|------------|
| LR         | 3.27       | 0.49    | 91.790     | 7.13      | 1.07    | 75.317     |
| OGD        | 2.71       | 0.43    | 91.843     | 6.52      | 0.95    | 76.072     |
| PA         | 6.48       | 0.43    | 91.791     | 13.88     | 0.94    | 75.195     |
| FOGD       | 5.46       | 1.07    | 96.172     | 180.97    | 42.36   | 85.316     |
| NOGD       | 92.45      | 8.17    | 92.468     | 336.65    | 29.78   | 71.721     |
| RRF        | 8.78       | 1.21    | 96.030     | 2936.24   | 214.44  | 85.045     |
| PAMO       | 107.18     | 20.93   | 96.258     | 101.66    | 10.62   | 81.204     |
| AVM        | 49.50      | 2.83    | 95.107     | 1418.75   | 7.51    | 71.149     |
| MPU-FOGDU  | 11.42      | 1.76    | 96.588     | 1353.59   | 20.21   | 90.880     |
| MPU-FOGDUB | 37.39      | 5.59    | 96.701     | 1419.69   | 19.91   | 90.927     |

Table 2: Comparison of training time, test time and test accuracy on the binary classification tasks.
| Algorithm      | combined          | mnist600k          |
|---------------|-------------------|-------------------|
|               | Train(s) | Test(s) | TestAcc(%) | Train(s) | Test(s) | TestAcc(%) |
| LR            | 3.25      | 0.17    | 73.563     | 22.73    | 1.04    | 53.961     |
| OGD           | 2.85      | 0.17    | 77.792     | 19.53    | 1.04    | 83.743     |
| PA            | 3.49      | 0.17    | 75.936     | 23.00    | 1.04    | 82.630     |
| FOGD          | 8.60      | 1.52    | 79.249     | 101.60   | 20.88   | 78.585     |
| NOGD          | 38.68     | 3.98    | 71.526     | 194.88   | 22.40   | 78.585     |
| RRF           | 66.35     | 2.89    | 79.995     | 361.01   | 39.36   | 95.712     |
| AVM           | 10.59     | 2.04    | 60.601     | 55.85    | 9.92    | 84.399     |
| MPU-FOGDU     | 117.68    | 2.19    | 83.223     | 126.21   | 18.43   | 99.096     |
| MPU-FOGDUB    | 121.26    | 2.20    | 83.214     | 137.20   | 18.48   | 99.097     |

| Algorithm      | poker         | acoustic        |
|---------------|---------------|-----------------|
|               | Train(s) | Test(s) | TestAcc(%) | Train(s) | Test(s) | TestAcc(%) |
| LR            | 40.25     | 1.69    | 27.160     | 3.22     | 0.17    | 67.039     |
| OGD           | 38.41     | 1.68    | 31.614     | 3.66     | 0.17    | 65.489     |
| PA            | 45.80     | 1.68    | 28.331     | 3.66     | 0.17    | 65.489     |
| FOGD          | 207.22    | 37.45   | 48.089     | 9.04     | 1.55    | 67.658     |
| NOGD          | 281.95    | 26.91   | 52.290     | 31.47    | 3.91    | 67.500     |
| RRF           | 153.10    | 6.85    | 48.702     | 10.82    | 0.60    | 67.438     |
| AVM           | 70.21     | 11.22   | 44.396     | 8.92     | 1.60    | 58.972     |
| MPU-FOGDU     | 304.61    | 32.07   | 93.658     | 63.16    | 2.16    | 74.032     |
| MPU-FOGDUB    | 347.70    | 32.30   | 94.270     | 64.66    | 2.11    | 74.304     |

| Algorithm      | forest       | aloi           |
|---------------|--------------|----------------|
|               | Train(s) | Test(s) | TestAcc(%) | Train(s) | Test(s) | TestAcc(%) |
| LR            | 20.74     | 0.99    | 51.347     | 100.34   | 0.56    | 58.419     |
| OGD           | 19.41     | 1.00    | 71.210     | 65.01    | 0.51    | 75.879     |
| PA            | 22.78     | 0.99    | 69.334     | 83.80    | 0.54    | 75.674     |
| FOGD          | 110.52    | 22.19   | 75.685     | 82.65    | 3.30    | 86.694     |
| NOGD          | 192.75    | 23.73   | 67.913     | 129.73   | 5.38    | 52.180     |
| RRF           | 931.69    | 37.56   | 75.298     | 209.03   | 9.11    | 86.778     |
| AVM           | 55.65     | 10.02   | 63.478     | 134.60   | 22.06   | 27.301     |
| MPU-FOGDU     | 238.76    | 19.46   | 84.884     | 178.55   | 2.52    | 83.576     |
| MPU-FOGDUB    | 279.44    | 19.14   | 84.951     | 178.91   | 2.33    | 83.597     |

Table 3: Comparison of training time, test time and test accuracy on the multi-classification tasks.
3.1 Experimental Setups

For all the algorithms, we use k-folder cross-validation to tune the parameters. For LR and OGD, the step size range is selected from \{10^{(-7)}, 10^{(-6)}, \ldots, 10^{(-1)}\}, while for FA, it is chosen from \{2^{(-8)}, 2^{(-7)}, \ldots, 2^{2}\}. We vary the Gaussian kernel width in the range of \{2^{(-10)}, 2^{(-14)}, 2^{(-12)}, 2^{(-10)}, 2^{(-8)}, 2^{(-6)}, 2^{(-4)}, 2^{(-2)}, 2^{0}, \sqrt{2}, 2^{4}\}. For MPU-FOGD algorithm, \eta_1 is set to 100, \eta_2 is 0.1 and \eta_3 is chosen from \{10^{(-6)}, 10^{(-4)}, 10^{(-2)}, 10^{(-1)}\}. The reason why \eta_1 can be set to 100 is that compared with other algorithms, such as FOGD, its gradient is multiplied by one in \(D\). For other algorithms, in nonlinear situations, the step size of gradient descent \eta_i is set to 0.001. The features of data samples are normalized in the interval \([-1, 1]\). For the binary classification algorithms, \(D\) in the FOGD, RRF, and MPU-FOGD algorithms are selected from \{200, 300, 400, 500, 1000, 2000, 4000, 6000, 8000\}. The rank \(k\) in NOGD is set to 0.2\(B\), and the budget size \(B\) is chosen through a random search in range \{200, 400, 600, 800, 1000\}. For multi-class algorithms, except for AVM and NOGD algorithms whose maximum budget value is set to 500, other budget parameters are set the same range: \{200, 300, 400, 500, 1000, 2000\}. We randomly divide all of the data sets into a training set, a test set and a validation set, of which the test set and the validation set account for 20\% respectively, the remaining 60\% for training. In order to better evaluate the performance of the classifier, we use the test accuracy, training time and test time as the performance indicators.

3.2 Experimental Results and Analysis

The experimental results on twelve binary classification and multi-class classification data sets are listed in Tables 2 and 3, respectively.

From Tables 2 and 3, we find that compared with nonlinear learning algorithms, the linear learning algorithms hold faster training and test time. However, their accuracy is not satisfactory. For nonlinear learning algorithms, in term of test accuracy, in most cases, MPU-FOGD is higher than the other algorithms.

3.3 Wilcoxon Signed-ranks Test

We conduct the Wilcoxon signed-ranks test [Dem06] to check whether the proposed method is significantly better than the other algorithms. The test compares the performances of two algorithms \(a\) and \(b\) on multiple datasets. To run the test, we rank the differences in performances of two algorithms for each dataset. The differences are ranked according to their absolute values. The smallest absolute value obtains the rank 1, the second smallest gets the rank 2. In case of equality, average ranks are assigned. The statistics of the Wilcoxon signed-ranks test is defined as [Dem06, YTH14]:

\[
\begin{align*}
    z(a, b) &= \frac{T(a, b) - n(n + 1)/4}{\sqrt{n(n+1)(2n+1)/24}}, \\
    T(a, b) &= \min\{R^+(a, b), R^-(a, b)\}.
\end{align*}
\]

where \(n\) is the number of the datasets, \(T(a, b) = \min\{R^+(a, b), R^-(a, b)\}\). \(R^+(a, b)\) is the sum of the ranks for the datasets on which algorithm \(a\) outperforms algorithm \(b\), and \(R^-(a, b)\) means the sum of the ranks for the opposite. They are defined as follows:

\[
\begin{align*}
    R^+(a, b) &= \sum_{d_i > 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i), \\
    R^-(a, b) &= \sum_{d_i < 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i),
\end{align*}
\]

where \(d_i\) is the accuracy difference between algorithms \(a\) and \(b\) on the \(i\)-th experimental dataset, \(\text{rank}(d_i)\) is the rank value of the absolute \(d_i\).

We fix \(b\) as MPU-FOGDUB, and let \(a\) as one of compared algorithms. From Tables 2-3, based on formulas (13)-(14), we can obtain that \(z(\text{FOGD, MPU-FOGDUB})=z(\text{RRF, MPU-FOGDUB})=-2.16, z(\text{NOGD, MPU-FOGDUB})=z(\text{AVM, MPU-FOGDUB})=-3.06.\) This shows that in term of test accuracy, MPU-FOGDUB, similarly as MPU-FOGDU, is significantly better than the other algorithms with the significance level \(\alpha = 0.05\).
4 Conclusions and Feature Work

In this paper, a multi-parameter updating Fourier online descent gradient algorithms (MPU-FOGD) are proposed to deal with large scale nonlinear classification problems. Theoretical analysis is provided to guarantee that the proposed random feature mapping can give a lower bound. Experimental results on several benchmark data sets demonstrate that compared with the state-of-the-art Fourier feature mapping online learning algorithms, the proposed MPU-FOGD can have better test accuracy.

For very large data sets, distributed learning provides a way to solve privacy-protected problems. At the same time, it also settles the problem that a single machine cannot handle or needs to spend a lot of time. In the future work, we are going to apply the work in this paper to design large-scale distributed classification algorithms.

References

[BDPDP96] Adam Berger, Stephen A Della Pietra, and Vincent J Della Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

[CDK+06] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7(Mar):551–585, 2006.

[CGLL10] Fu Chang, Chien-Yang Guo, Xiao-Rong Lin, and Chi-Jen Lu. Tree decomposition for large-scale svm problems. *The Journal of Machine Learning Research*, 11:2935–2972, 2010.

[CHL08] Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. Coordinate descent method for large-scale l2-loss linear support vector machines. *Journal of Machine Learning Research*, 9(Jul):1369–1398, 2008.

[CTJ09] Haibin Cheng, Pang-Ning Tan, and Rong Jin. Efficient algorithm for localized support vector machine. *IEEE Transactions on Knowledge and Data Engineering*, 22(4):537–549, 2009.

[CYYX16] Xiaojun Chang, Yao-Liang Yu, Yi Yang, and Eric P Xing. Semantic pooling for complex event analysis in untrimmed videos. *IEEE transactions on pattern analysis and machine intelligence*, 39(8):1617–1632, 2016.

[Dem06] Janez Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

[DNSV19] Tejas Indulal Dhamecha, Afzel Noore, Richa Singh, and Mayank Vatsa. Between-subclass piece-wise linear solutions in large scale kernel svm learning. *Pattern Recognition*, 95:173–190, 2019.

[FBdCÁRJ+14] Ivsviis Frías-Blanco, José del Campo-Ávila, Gonzalo Ramos-Jimenez, Rafael Morales-Bueno, Agustín Ortiz-Diaz, and Yailé Caballero-Mota. Online and non-parametric drift detection methods based on hoeffding’s bounds. *IEEE Transactions on Knowledge and Data Engineering*, 27(3):810–823, 2014.

[FS01] Shai Fine and Katya Scheinberg. Efficient svm training using low-rank kernel representations. *Journal of Machine Learning Research*, 2(Dec):243–264, 2001.

[GJP18] Yiqing Guo, Xiuping Jia, and David Paull. Effective sequential classifier training for svm-based multitemporal remote sensing image classification. *IEEE Transactions on Image Processing*, 27(6):3036–3048, 2018.

[GMCR04] João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. In *Brazilian symposium on artificial intelligence*, pages 286–295. Springer, 2004.
Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S Sathiya Keerthi, and Sellamanickam Sundararajan. A dual coordinate descent method for large-scale linear svm. In Proceedings of the 25th international conference on Machine learning, pages 408–415, 2008.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. 58:13–30, 1963.

S Hare, A Saffari, and PH Torr. S. 2016. struck: Structured output tracking with kernels. IEEE Transactions on Pattern Analysis & Machine Intelligence, 38(10).

Steven CH Hoi, Jialei Wang, and Peilin Zhao. Libol: A library for online learning algorithms. The Journal of Machine Learning Research, 15(1):495–499, 2014.

Thorsten Joachims. Training linear svms in linear time. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 217–226, 2006.

Javier Jorge and Roberto Paredes. Passive-aggressive online learning with nonlinear embeddings. Pattern Recognition, 79:162–171, 2018.

S Sathiya Keerthi and Dennis DeCoste. A modified finite newton method for fast solution of large scale linear svms. Journal of Machine Learning Research, 6(Mar):341–361, 2005.

Bin Kang, Dong Liang, Wan Ding, Huiyu Zhou, and Wei-Ping Zhu. Grayscale-thermal tracking via inverse sparse representation-based collaborative encoding. IEEE Transactions on Image Processing, 29:3401–3415, 2019.

S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karunthi Radha Krishna Murthy. Improvements to platt’s smo algorithm for svm classifier design. Neural computation, 13(3):637–649, 2001.

Jing Lu, Steven CH Hoi, Jialei Wang, Peilin Zhao, and Zhi-Yong Liu. Large scale online kernel learning. The Journal of Machine Learning Research, 17(1):1613–1655, 2016.

Yi Li and Philip M Long. The relaxed online maximum margin algorithm. In Advances in neural information processing systems, pages 498–504, 2000.

Yuh-Jye Lee and Olvi L Mangasarian. Rsvm: Reduced support vector machines. In Proceedings of the 2001 SIAM International Conference on Data Mining, pages 1–17. SIAM, 2001.

Trung Le, Tu Nguyen, Vu Nguyen, and Dinh Phung. Dual space gradient descent for online learning. In Advances in Neural Information Processing Systems, pages 4583–4591, 2016.

Trung Le, Tu Dinh Nguyen, Vu Nguyen, and Dinh Phung. Approximation vector machines for large-scale online learning. The Journal of Machine Learning Research, 18(1):3962–4016, 2017.

Quoc V Le, Alex J Smola, and Svn Vishwanathan. Bundle methods for machine learning. In Advances in neural information processing systems, pages 1377–1384, 2008.

Ming Lin, Shifeng Weng, and Changshui Zhang. On the sample complexity of random fourier features for online learning: How many random fourier features do we need? ACM Transactions on Knowledge Discovery from Data (TKDD), 8(3):1–19, 2014.

Olvi L Mangasarian and David R Musicant. Lagrangian support vector machines. Journal of Machine Learning Research, 1(Mar):161–177, 2001.
[NGB17] Loris Nanni, Stefano Ghidoni, and Sheryl Brahnam. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158–172, 2017.

[NLBP17] Tu Dinh Nguyen, Trung Le, Hung Bui, and Dinh Q Phung. Large-scale online kernel learning with random feature reparameterization. In *IJCAI*, pages 2543–2549, 2017.

[PV16] Ali Pesaranghader and Herna L Viktor. Fast hoeffding drift detection method for evolving data streams. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 96–111. Springer, 2016.

[Ros58] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386–408, 1958.

[RR08] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

[SS00] Alex J Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. 2000.

[SSSS07] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007)*, Corvallis, Oregon, USA, June 20-24, 2007, 2007.

[TKC05] Ivor W Tsang, James T Kwok, and Pak-Ming Cheung. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, 6(Apr):363–392, 2005.

[TVSL10] Choon Hui Teo, SVN Vishwanathan, Alex Smola, and Quoc V Le. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11(1), 2010.

[WSH+18] Zeyi Wen, Jiashuai Shi, Bingsheng He, Jian Chen, and Yawen Chen. Efficient multiclass probabilistic svms on gpus. *IEEE Transactions on Knowledge and Data Engineering*, 31(9):1693–1706, 2018.

[WZH+13] Jialei Wang, Peilin ZHAO, Steven CH Hoi, Jinfeng Zhuang, and Zhi-yong Liu. Large scale online kernel classification. 2013.

[XZL+18] Jun Xu, Wei Zeng, Yanyan Lan, Jiafeng Guo, and Xueqi Cheng. Modeling the parameter interactions in ranking svm with low-rank approximation. *IEEE Transactions on Knowledge and Data Engineering*, 31(6):1181–1193, 2018.

[YDL19] Rui Ye, Qun Dai, and MeiLing Li. A hybrid transfer learning algorithm incorporating trsvm with gasen. *Pattern Recognition*, 92:192–202, 2019.

[YSZ+18] Yazhou Yao, Fumin Shen, Jian Zhang, Li Liu, Zhenmin Tang, and Ling Shao. Extracting privileged information for enhancing classifier learning. *IEEE Transactions on Image Processing*, 28(1):436–450, 2018.

[YTH14] Xiaowei Yang, Liangjun Tan, and Lifang He. A robust least squares support vector machine for regression and classification with noise. *Neurocomputing*, 140:41–52, 2014.

[ZBMM06] Hao Zhang, Alexander C Berg, Michael Maire, and Jitendra Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*, volume 2, pages 2126–2136. IEEE, 2006.

[Zha04] Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, page 116, 2004.
[Zin03] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 928–936, 2003.

[ZOW16] Rui Zhao, Wanli Oyang, and Xiaogang Wang. Person re-identification by saliency learning. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):356–370, 2016.

[ZSW+19] Lihe Zhang, Jiayu Sun, Tiantian Wang, Yifan Min, and Huchuan Lu. Visual saliency detection via kernelized subspace ranking with active learning. *IEEE Transactions on Image Processing*, 29:2258–2270, 2019.

[ZWZ+17] Wangmeng Zuo, Faqiang Wang, David Zhang, Liang Lin, Yuchi Huang, Deyu Meng, and Lei Zhang. Distance metric learning via iterated support vector machines. *IEEE Transactions on Image Processing*, 26(10):4937–4950, 2017.

[ZZH+16] Zhaoze Zhou, Wei-Shi Zheng, Jian-Fang Hu, Yong Xu, and Jane You. One-pass online learning: A local approach. *Pattern Recognition*, 51:346–357, 2016.