An Adaptive Classifier with a Nonparametric Local Filter for Brain-Computer Interface

Qin Jiang, Yi Zhang, Gengyu Ge, and Wei Wang

***Abstract*—A potential limitation of motor imagery (MI) based brain-computer interface (BCI) is that it usually takes a long time to record sufficient electroencephalogram (EEG) data for a robust classifier. Besides, the well-trained classifier has weak adaptability in cross-session or sample-wise online testing. We propose an adaptive classifier with a nonparametric filter to tackle this problem by incorporating test data into the self-training process to alleviate the over-fitting and non-stationarity in EEG. In the framework, we exploit the natural neighbor to explore the underlying structure of labeled and unlabeled data, then introduce a novel neighborhood editing filter (NENaN) to measure the contribution of an instance to the classifier. We focus on LDA as a base classifier. Comparative studies with several representative filters in the literature on three datasets demonstrate that NENaN is outstanding in filtering out mislabeled samples and outliers, and has advantages in improving the performance of an adaptive classifier.**

***Index Terms*—Adaptive classifier, Self-training method, Natural Neighbor, Motor imagery, Brain-computer interface**

I. Introduction

BRAIN-Computer Interface (BCI) is a human-computer interaction technology that establishes a direct information pathway between a human brain and external equipment without relying on the peripheral nerve and muscle system[1][2][3]. It has demonstrated prospects in the rehabilitation and assistance of disabled people such as intelligent wheelchairs, artificial limbs, spelling systems [4] [5] [6] [7], etc. Motor imagery (MI) is prized for its ease of set-up, simple stimulus, and closer to the human being's natural thinking way [8][9], which has become an emerging research area.

Brain-Computer Interface based on motor imagery (MI-BCI) has not been applied in practice despite extensive research. Generally, the main challenges faced by online MI-BCI system are the limited amount of training data and non-stationarity in EEG over time, which may induce poor robustness and generalization ability of the offline model, and poor adaptability in the testing phase [10] [11] [12].

The semi-supervised adaption was developed to deal with limited training data by employing both labeled data and incoming unlabeled data to accommodate a classifier, thus requiring fewer offline data [13] [14] [15]. Clustering analysis provides an option for mining unlabeled data distribution. Wu [16] discovered the structure of unlabeled data by finding density peaks of data and then integrated the structure space into the self-training process to train a classifier iteratively. Gan [17] combined the fuzzy c-means algorithm into self-training classification to introduce a better SVM classifier. In [18], the co-labeling method with a random forest was adopted to enhance the performance of a self-training classifier degraded by insufficient labeled data. However, in the process of self-training, the adaptive classifier may treat the mislabeled or noisy samples as confident instances and incorporate them into the train set [21], which worsens the classifier's robustness.

To effectively remove outliers or noise from an unlabeled dataset, Triguero [22] focused on the nearest neighbor as a base classifier and analyzed the integration of a wide variety of noise filters into the self-training process to distinguish the most relevant features, such as ENN, SETRED [23] and CEWS. However, those mentioned filters all take KNN as the primary classifier, whose parameter significantly influences the filter. Zhu [24] proposed a parametric clustering algorithm, i.e., natural neighbor clustering (NaN), compared with KNN, K-means [25], and density peaks clustering (DPC), NaN is not sensitive to the selection of clustering center and parameters [23] [25]. Based on NaN, Huang [26] introduced a natural outlier factor to measure the outliers. Yang [27] presented a natural neighborhood graph-based instance reduction algorithm, which exploited a natural neighborhood graph to divide the data into noisy, boundary point, and core instances. Li [28] introduced a novel self-training method based on density peaks with a parameter-free local to removed mislabeled samples. Other outlier detection algorithms, such as LOF [29], INFLO [30], and INS [31], also show effectiveness in outlier filtering. However, the above noise filters either evaluate the density of the sample distribution, or measure the label's frequency, which do not perform well when the labeled data are insufficient.

Moreover, semi-supervised adaptive classifiers require full retraining with new incoming data. Most existing semi-supervised methods use SVM [12] [14-18] or KNN as base classifier [26] [29] [30], which are time-consuming for the parameter-dependent character [19] [20]. Herein we prefer the LDA without hyperparameters as the basic classifier in this paper.

Manuscript received Dec. 8, 2020; revised Jul. 13, 2021. This work was supported by the National Nature Science Foundation of China (No. 51775076) and the Doctoral Program of Chongqing University of Posts and Telecommunications (BYJS201910).

Qin Jiang is a Ph.D. candidate of School of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, CO 400065 China (e-mail: namy_jiang@hotmail.com).

Yi Zhang is a Professor of Advanced Manufacturing and Automatization Engineering Laboratory, Chongqing University of Posts and Telecommunications, Chongqing, 400065 China (the corresponding author, phone: 023-65261578; fax: 023-65264781; e-mail: zhangyi@cqupt.edu.cn).

Gengyu Ge is a Ph.D. candidate of Chongqing University of Posts and Telecommunications, Chongqing, 400065 China (e-mail: D190201004 @stu.cqupt.edu.cn).

Wei Wang is a Ph.D. candidate of Chongqing University of Posts and Telecommunications, Chongqing, 400065 China (e-mail: D190201021 @stu.cqupt.edu.cn).

In the light of background, a nonparametric local outlier filter, neighborhood editing based on natural neighbor (NENaN), is proposed to evaluate the newly added data by exploiting labels and distribution information. NENaN divides the data into four categories: mislabeled instances, isolated instances, boundaries, and supporting instances. Isolated points are far from cluster centers, borders near the hyperplane, and supporting instances around cluster centers. A self-training classifier based on NENaN can efficiently track changes in EEG over time by incrementally re-estimating and updating parameters with borders and supporting. Moreover, we present the local representative (LR) concept to reduce the size of supporting samples to decrease training time and computation complexity, which is of great significance to an online BCI system.

In brief, the main contributions of our work include:

1)We propose an adaptive LDA classifier based on semi-supervised learning to mitigate non-stationarity in EEG.

2)We integrate a nonparametric local filter into the LDA self-training process, which is also beneficial to initialize the classifier.

3)We introduce the local representative (LR) concept to shrink the scale of supporting instances to reduce the computation complexity and storage consumption in the self-training process.

4)The feasibility of our adaptive classifier framework is verified in discrete and continuous BCI datasets.

The rest paper proceeds as follows: Section II introduces related works, and Section III details our proposed algorithm and its implementation framework. Section IV provides a detailed description of the experiment design and results on three datasets. Finally, section V draws a conclusion and directions for future work.

II. Background

A. Notations

$x_i \in R^m$: a m -dimension feature vector.

$L = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$: the samples set with labels.

$U = \{x_1^u, x_2^u, ..., x_t^u\}$: the set without labels.

$Nb(x)$: the number of natural neighbors of x .

$NaN_k(x)$: the natural neighbors set of x .

$NN_k(x)$: k -th nearest neighbors set of x .

$RNN_k(x)$: k -reverse nearest neighbors set of x .

$dist(x_i, x_j)$: the Euclidean distance between x_i and x_j .

B. Natural Neighbor

The concept of natural neighbor is inspired by the human social relationship that we call a friend if we know each other. Similarly, B is a natural neighbor of A, only if A is also a natural neighbor of B. Although the natural neighbor relationship derives from the k -th nearest neighbor (KNN) searching [24] [27], there are differences. KNN establishes a neighborhood by actively searching its k nearest neighbors, whereas NaN is a byproduct of KNN, and its procedure is entirely passive. Besides, KNN relies greatly on the parameter k , whereas NaN, without setting any parameters, automatically searches based on the distribution of feature space until a natural stable structure is formed. The NaN searching procedure for A, B and C is stated in Fig.1.

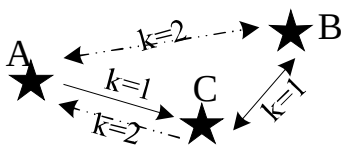


Fig. 1. The schematic of a natural neighbor search.

We first conduct a nearest neighbor search with $k = 1$ for the three samples, respectively, where we get:

$$\begin{cases} NN_1(A) = C \\ NN_1(B) = C \Rightarrow NaN(B) = C \\ NN_1(C) = B \end{cases}$$

A has no natural neighbors yet, therefore, the searching continues with $k = 2$:

$$\begin{cases} NN_2(A) = B \\ NN_2(B) = A \\ NN_2(C) = A \end{cases} \Rightarrow \begin{cases} NaN(B) = A \\ NaN(A) = C \end{cases}$$

When $k = 3$, the natural neighbor relationship of the three objects no longer changes; let's say that the three have established a stable natural structure. It is clear, throughout the searching process, the association of natural neighbors needs no specific k value.

Definition 1 (Natural Neighbor). If x_i is the k-th nearest neighbor of x_j, and x_j is the r-th nearest neighbor of x_i, x_i is a natural neighbor of x_j.

$$x_i \in NaN(x_j) \Leftrightarrow x_j \in NN_k(x_i) \&\& \, x_i \in NN_r(x_j) \quad (1)$$

Definition 2 (Natural Stable Structure). Suppose each object has at least one natural neighbor, and the number of neighbors remains unchanged with the expansion of the search scope. In such case, the set is considered to have formed a relatively stable state, called a natural stable structure.

$$\forall x_i, Nb(RNN_k(x_i)) = Nb(RNN_{k-1}(x_i)) \quad (2)$$

Definition 3 (Natural Eigenvalue). The natural eigenvalue is the minimum k when reaching a natural stable structure, marked as $\sup_k$.

$$\sup_k(x_i) = \min\{r \mid \forall x_i \exists x_j (x_i \neq x_j \cap x_i \in NN_r(x_j))\} \quad (3)$$

The natural neighbor searching pseudo-code is demonstrated in Algorithm 1.

Algorithm 1: Natural Neighbor searching (NaNSearch)

Input: Training dataset $X(x_i \in X)$

Output: $NN_{k,\gamma}$

1.Initialization $k = 1$, $Nb(x_i) = \phi$, $NN_k(x_i) = \phi$, $RNN_k(x_i) = \phi$

2. create a k-d tree T of dataset X

3. for each sample find x_i its γ -th neighbor x_j by T

$NN_k(x_i) = NN_k(x_i) \cup x_j$

$RNN_k(x_j) = RNN_k(x_j) \cup x_i$

$Nb(x_j) = Nb(x_j) + 1$

end for

5.compute the number of satisfy $Nb(x_i) = 0$

6.if the num does change

return NN_k, γ

else

$\gamma = k + 1$

goto step 3

end if

C. Local Representative

The LR aims at finding representatives of neighborhoods to simplify the complex manifold structure [26].

Definition 4 (Local Representative). x_i is a local representative for $NaN_k(x_j)$, if x_i is a natural neighbor of $NaN_k(x_j)$ and the density of x_i is greatest in $NaN_k(x_j)$.

$$LR(x_i) = NaN_k(x_j) \Leftrightarrow x_i \in NaN_k(x_j) \\ \&\&(\rho_{x_i} \geq \rho_{x_k}, \forall x_k \in NaN_k(x_j)) \tag{4}$$

In $NaN_k(x_i)$,the density of x_i is defined as:

$$\rho_i = \frac{\sup_k}{\sum_{x_j \in NaN_k(x_i)} (dist(x_i, x_j))} \tag{5}$$

Definition 5 (Representative Transfer Rule (RTR)). If $LR(x_1) = NaN_k(x_j)$, $LR(x_2) = NaN_k(x_k)$, and $x_2 \in NaN_k(x_j)$ then $LR(x_1) = NaN_k(x_k)$.

D. LDA

LDA has the advantages of a simple model, less computation cost, and high efficiency in binary classification. LDA optimization is to make the inter-class distance large, whereas the intra-class distance is as close as possible. Its objective function is to solve a generalized Rayleigh quotient without hyperparameters, making LDA very applicable as the basic classifier for an online BCI system. The LDA classifier can be expressed as:

$$\begin{cases} f(x_i) = \omega^T x_i + b \\ b = -\omega^T (\mu_1 + \mu_2)/2 \\ \omega = \sum^{-1} (\mu_1 - \mu_2) \end{cases} \tag{6}$$

where ω , b are projection vector and bias of the optimal segmentation hyperplane, respectively. μ_c is the c-th class mean($c \in \{1, 2\}$). $\sum = \sum_1 + \sum_2$, $\sum_c$ is c-th covariance matrix. If the observation x_i satisfies $f(x_i) > 0$, it is classified as class 1, otherwise, as class 2.

Fig. 2 demonstrates the first 100trials feature distribution of subject 'AA'(see Section IV, Dataset2) extracted by one pair of Common Spatial Pattern (CSP) spatial filer. Three types of samples are marked, mislabeled (indicated as 1,2), isolated (as 3,4), and overlapped samples (as 5,6). Isolated instances increase the intra-class mean, which defeats the optimization goal of LDA. Removing mislabeled instances facilitates accuracy, whereas removing overlapping instances has little impact on the inter or intra-class mean. The LDA with a NENaN filter can effectively overcome the deviation by eliminating isolated and mislabeled instances from the dataset.

Moreover, due to non-stationary factors such as fatigue, emotion, and psychological state, the feature distribution between the train and test dataset shows significant fluctuation. The classifier trained offline may have limited generalization ability in the testing phase, especially for a long-duration test. Thus, an adaptive classifier combined with a NENaN filter is conductive to relieve feature drift.

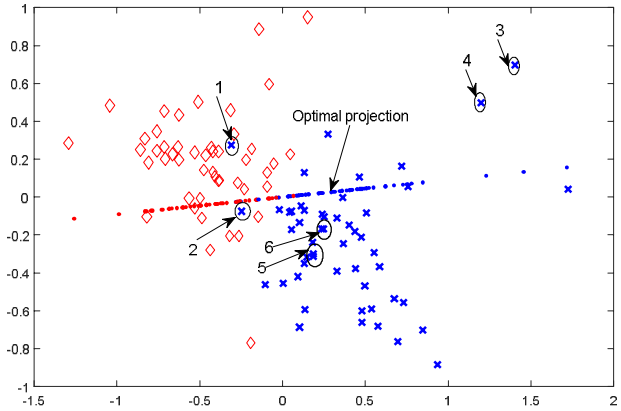


Fig.2. The feature space and its projection vector.

III. Proposed Algorithms

In this section, we detail the framework of an adaptive classifier based on NENaN. As shown in Fig.3, the flowchart consists of two parts: offline training and online testing. In the offline stage, the NENaN first divides the offline dataset into four categories: noise, outliers, supporting samples, and boundaries, and then an LDA classifier is initialized by the supporting and boundary instances. Moreover, we replace the large-scale supporting samples with a small number of representatives with local maximum density, resulting in a core dataset consisting of representatives and boundary points. In the online phase, NENaN evaluate the test data by working with the obtained core set, where boundaries are used to re-estimate a classifier, and the core set is preserved for another round of assessment.

A. Structure Discovery Based on NaN

It figures out from Fig.2 that mislabeled samples and outliers will sharpen LDA deviation. In contrast, supporting samples reflect the overall distribution and determine the optimal projection direction, and the boundary samples determine the bias. Therefore, it is necessary to eliminate samples with adverse effects. Clustering is a conventional method for structure analysis, such as KNN, DPC, etc., but it is difficult to determine appropriate parameters [24] [26] [27]. NaN automatically clusters according to the distribution of samples without any parameters. See algorithm 1 for the detailed process. The termination criteria for algorithm 1 is that the most isolated sample contains at least one k-th neighbor. However, this standard of linking each point to its $\min\{Nb(x_i), \sup_k\}$ nearest neighbors

does not guarantee a stable neighbor relationship, i.e. if x_i is a neighbor of x_k, x_k is not necessarily a neighbor of x_i. So, we improve the termination criteria to make the most isolated sample has at least one natural neighbor. According to this criterion, each point will relate to $\max\{Nb(x_i)\}$ neighbors to form a relatively stable neighbor relationship. A density-based detection algorithm can be employed to find outliers without any parameters in a stable neighbor relationship. However, actual experiments demonstrate that the natural eigenvalue will exceed the number of samples, resulting in substantial time-consuming when outliers are considerably away from cluster centers. For this reason, the restriction is added $Nb(x_i) < N/2$. Algorithm 2 is an improved NaN search algorithm.

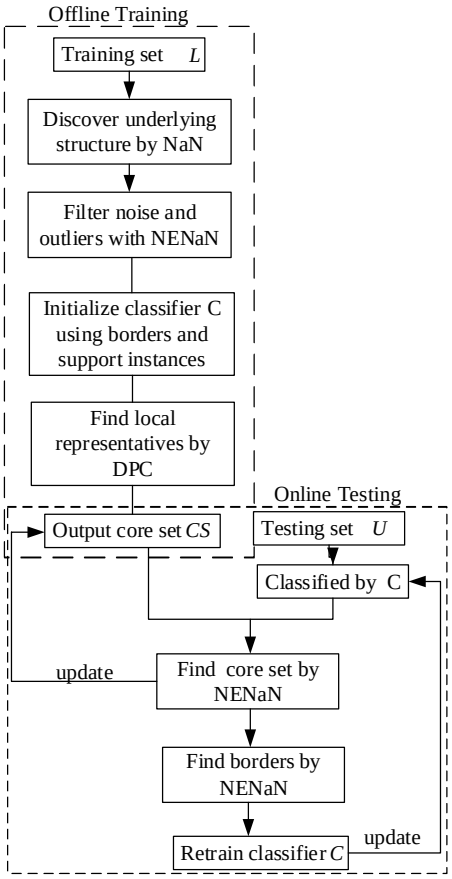


Fig.3. the framework of proposed adaptive classifier

Algorithm 2: Improved NaN searching (INaNSearch)

Input: Training data $x_i \in R^m, (i = 1, 2, ...N)$

Output: NaN_k, γ

1.Initialization $k = 1$, $Nb(x_i) = \phi$, $NN_k(x_i) = \phi$, $RNN_k(x_i) = \phi$, $NaN_k(x_i) = \phi$

2. create a k-d tree T of data set X ($x_i \in X$)

3. for each sample find x_i its γ -th neighbor x_j by T

$NN_k(x_i) = NN_k(x_i) \cup x_j$

$RNN_k(x_j) = RNN_k(x_j) \cup x_i$

end for

4. for each sample find x_i

$NaN_k(x_i) = RNN_k(x_i) \cap NN_k(x_i)$

end for

5.compute the number of neighbors in $NaN_k(x_i)$

$Nb(x_i) = num(NaN_k(x_i))$

6.if each $Nb(x_i)! = 0$ or $\gamma \geq N/2$

$\gamma = k$

return NaN_k, γ

else

$\gamma = k + 1$

goto step 3

end if

B. Neighborhood Editing Based on Natural Neighbor

NENaN edits each neighborhood generated by the natural neighbor, and the type and number of labels in $NaN_k(x)$ reflect the difference between the query and its neighbors. The NENaN addresses how to quantify such difference and employ the quantized difference to distinguish border, core, and mislabeled samples.

In the filters based on KNN (such as ENN, RENN, ALLKNN, MLSTE) [21] or NaN (such as NNGIR[27], STDFNF[28]), a noisy instance is determined by whether the identical frequency is more than half of all neighbors. However, it is rough to evaluate a sample only based on the frequency of identical tags in the neighborhood. Fig.4, as an example, explained this view.

Fig.4 depicts 10 neighbors of query 'A'. Although the number of positive instances (denoted by '●') is less than that of negative ones (denoted by '★'), positive neighbors in the neighborhood are relatively denser than the distribution of negative instances, so 'A' is more likely to be positive.

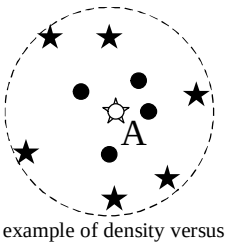


Fig.4. An example of density versus frequency

In NeNaN, we consider the first and second echelon neighbors and their labels to quantify the confidence of a query instance. The first echelon members consist in the domain split by the first 'Target-Others' line, and the second echelon refers to the domain cut by the second 'Target-Others' line. The 'Target-Others' is the cut-off line, where the tag of the current instance differs from that of the following one. We defined two indexes, distribution sparsity β and tag tendency γ, as expressed in (7).

$$\begin{cases} \beta = \dfrac{fHom(x) + 1/\sup_k}{Het(x) + 2/\sup_k} \\ \gamma = \dfrac{fHom(x) + sHom(x)}{\sup_k} \end{cases} \tag{7}$$

Definition 6 (the first homogeneousness of x, $fHom(x)$). In the set composed of x and its natural neighbors, $fHom(x)$ is the number of elements with the identical label as x in the first echelon domain.

$$fHom(x) = |\{z \mid z \in (NaN(x) \cup x) \& l(z_{i+1}) = l(z_i) = l(x)\}| \tag{8}$$

$sHom(x)$ denotes the number of elements consistent with the label of x in the second echelon domain.

Definition 7 (heterogeneousness of x, $Het(x)$). $Het(x)$ is the number of elements with labels different from x in the first echelon domain.

$$Het(x) = |\{z \mid z \in (NaN(x) \cup x) \& l(z_{i+1}) = l(z_i), l(z_i) = l(x)\}| \tag{9}$$

where $|.|$ indicates the number of a set, $l(.)$ is the label of the sample.

β is the ratio of $fHom(x)$ and $Het(x)$, representing the distribution density of a query among its near neighbors. When $\beta < 1/2$, the label of the query is the minority in its nearest neighbors, which has the risk of misclassification, otherwise, when $\beta \geq 1/2$, the larger the β is, the closer the query instance is to its cluster center, and the higher reliability of the prediction is.

γ is the ratio of the number of samples with the same label as the query to the natural eigenvalue, indicating the probability of the query label converting to other types. When $\gamma < 1/2$, the current tag is quite possible to shift to another type as the neighborhood expands, conversely, the current type tends to remain constant, when $\gamma \geq 1/2$. The combination of two indicators can evaluate the prediction confidence of a test sample.

Definition 8 (mislabeled sample, (MS)). If an instance satisfies $\beta < 1/2$ and $\gamma < 1/2$, it is misclassified, i.e. the label of query instance is different from that of its near neighbors, and with the expanding of neighborhood, it tends to transfer to other categories.

Definition 9 (isolated sample, (IS)). The isolated instance are far from the cluster center and have at most one natural neighbor, subjecting to $\beta > 1$ and $\gamma < 1/2$.

Definition 10 (boundary sample, (BS)). A border satisfies $1/2 \leq \beta \leq 1$, i.e., the natural neighborhood of a border contains different categories, but the category to which it belongs predominates.

Definition 11 (supporting sample, (SS)). The category of the query is absolutely dominant in its nearest neighbors, and this advantage will not change with the domain expansion, i.e., a core instances satisfy $\beta > 1$ and $\gamma \geq 1/2$.

Fig.5 depicts four examples for the query instance based on the natural neighbors editing.

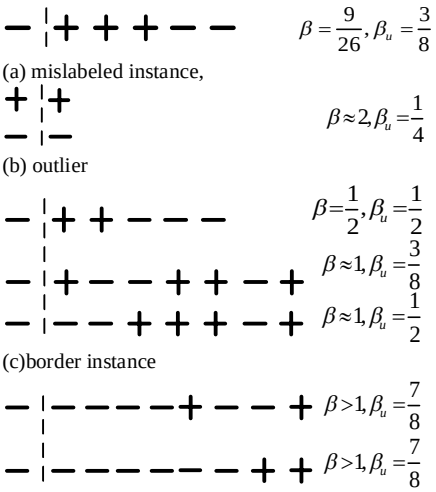


Fig.5. four examples of natural neighborhoods, where sup_k = 8, and the left of the dotted line is the prediction label of the query sample,'+' represents a positive instance, and '-' represents a negative one.

By employing the NENaN filter, we can filter out noisy instances and outliers, and initialize an LDA classifier with supporting samples and boundaries. Algorithm 3 details the process of a Neighborhood Editing filter based on NaN.

Algorithm 3: Neighborhood Editing on NaN (NENaN)

Input: NaN_k, γ and training dataset L

Output: BS, SS

1.Initialization $BS = \phi$, $NS = \phi$, $IS = \phi$, $SS = \phi$

2. calculate $HetP(x)\, HomP(x)$

3. for each sample x_i in L

  if $\beta > 1$ && $\gamma < 1/2$

    $IS = IS \cup \{x_i\}$

  else if $\beta < 1/2$ && $\gamma < 1/2$

    $MS = MS \cup \{x_i\}$

  else if $1/2 \leq \beta \leq 1$

    $BS = BS \cup \{x_i\}$

  else if $\beta > 1$ && $\gamma \geq 1/2$

    $SS = SS \cup \{x_i\}$

  end if

end for

4. $SS = setdiff(SS, IS)$ $BS = setdiff(BS, IS)$

$BS = setdiff(BS, SS)$

The time complexity of NENaN is $O(Nlog(N))$, where N is the number of the current train set and unlabeled samples. However, with the incremental learning going, the train set expands continuously, which increases the time consumption of natural neighbor clustering. To this end, we replace the supporting samples with fewer representatives. According to definition 4, we need to calculate the represent-

atives of each NaN(x) using (5), and then get a representative set of the supporting set according to the RTR rule (definition 5). The LR method reduces the size of the train set while maintaining the distribution characteristics of the core set.

Algorithm 4: LDAbasedonNENaN (NELDA)

Input: NaN_k, L, U
Output: the retrained LDA classifier C
Initialization:
1. $[NaN_k, \gamma]$=**INaNSearch(**L**)**
2. $[BS, SS]$=**NENaN(**NaN_k, γ, L**)**
3. CS =**LRsearch(**NaN_k, BS, SS**)**
% LR searching, CS is the core set for train set
4.Train an initial LDA classifier C with $[BS, SS]$
Begin:
Repeat until the online testing process is complete
5. label U using the trained classifier C
6. BS =**eNENaN(**CS, U**)%** extended NENaN
7.Update u_c, Σ according (2) and (4) using BS
8.retrain classifier C using BS

C. Adaptive LDA Classifier with NENaN

The adaptive classifier incorporates new coming data to re-estimate parameters to adapt to the current feature space. Equation (1) illustrates that the classifier is determined by global covariance $\sum$ and class mean μ_c. In this paper, we update $\sum$ and μ_c in a supervised way wherein NENaN assesses the prediction confidence of the new data.

We assume $\mu_c(t-1)$ as the class mean of previous iteration, and the current mean $\mu_c(t)$ can be inferred by the following formula:

$$\mu_c(t) = (1-\eta).\mu_c(t-1) + \eta.G_c$$
$$G_c = \frac{1}{n}\sum_{i=1}^{n} x_c^i(t) \tag{10}$$

where $x_c^i(t)$, $i = \{1, 2, ...n\}$ denotes the current new coming instances of c -class. G is global mean of new data, and G_c denotes class mean of c -class new data. η is an update coefficient a compromise between the update rate and the system robustness.

Here, the inverse of covariance matrix $\sum(t)$ can be recursively estimated by the Sherman-Morrison-Woodbury theorem, which avoids roughly complicated matrix inverse.

$$\sum(t)^{-1} = \frac{1}{(1-\eta)} * \left[\sum(t-1)^{-1} - \frac{\sum(t-1)^{-1} G G^T (\sum(t-1)^{-1})^T}{\frac{1-\eta}{\eta} + G^T \sum(t-1)^{-1} G} \right] \tag{11}$$

By (10) and (11), the covariance $\sum$ and mean vector μ_c are estimated online, and then substituted into (6) to retrain the LDA classifier. Algorithm 4 is a self-training process based on NENaN.

IV. Experiments

A. Dataset Description and Experiment Design

1) BCI competition IV Dataset 2b [32] (Dataset1) was collected using 3 bipolar active electrodes C3, Cz, C4 at a sampling rate of 250Hz from 9 subjects (noted as B01~B09). Each subject performed left-and right-handed motor imagery for 5 sessions. The first two sessions comprise 120 trials per session without feedback, while the last three sessions are 160 trials per session with feedback.

2) BCI Competition III Dataset IVa [33] (Dataset2) contains EEG signals recorded at 118 channels with a 1000Hz sampling rate (downsampled to 100Hz in this paper) from five subjects (named as 'AA''AY''AW''AL''AV'). For each subject, a total of 280 cue-based trials are available (half per MI task). In each trial, a cue was indicated for 3.5s, during which two MI tasks were performed: (R) right hand, (F) right foot. Then the cue was intermitted by periods of random length, 1.75 to 2.25s, in which the subject could relax. Herein eight active electrodes in the motion region FC3, FC4, C3, C1, C4, C2, CP3, CP4 were selected for analysis.

3) BCI Competition III Dataset V [34] (Dataset3) is continuous EEG for three cued mental imagery, left-hand movement, right-hand movement and word generation. Each subject performed 4 sessions, each lasting 4 minutes with 5-10 minutes breaks in between them. The subject performed a given task for about 15seconds and then switched randomly to another task. Our work employed precomputed features dataset that raw EEG signals were computed power spectral density (PSD) 16 times per second in the band 8~30Hz.

Dataset1 and Dateset2 both were filtered by a 4-order Butterworth bandpass filter of 8-30Hz with bandpass attenuation 0.5dB. In Dataset1, data of 4~6s was selected as effective motor imagery duration of each trial, and a 3-second segment was captured after the cue appearing 0.5s in Dataset2. We extracted the power spectral density (PSD) of α, β bands as features, and each channel 12 frequency components (2Hz frequency resolution). A 0.5-second sliding window captured Dataset3 without overlapping, and the average of 8 consecutive samples was regarded as feature, i.e., the BCI system output every 0.5 seconds.

We designed three experiments to verify our algorithm framework: Experiment I demonstrated the advantage of NENaN in initializing a LAD classifier, and performed the selection of η; Experiment II confirmed the merits of the proposed adaptive LDA by comparing NENaN with exist state-of-art filters; Experiment III explored the effectiveness of NENaN in the multi-category continuous motion imagery scenario, with SVM, KNN and LDA being the base classifier, respectively. Experiment I and II were conducted on Dataset1and 2, Experiment III on Dataset3. Table I summarizes a detailed description of these datasets.

TABLE I
DESCRIPTION OF EXPERIMENTAL DATA SETS

No.	Data sets	Size	Attribute	Class
1	BCI Competition IV-2b	720	36	2
2	BCI Competition III-IVa	280	96	2
3	BCI Competition III-V	1754/1734/ 1722	96	3

The specific settings of each experiment are as follows:

1)Experiment I. Random half of Dataset1 was used for training, the rest for evaluation, and three sessions of Dataset2 for training and the rest for testing. For an optimal η, LDA was retrained according to (10) and (11) with five-fold cross-validation on testing data, η ranging in $\{0.01, 0.1, 0.2, 0.3, 0.4, 0.5\}$.

2)Experiment II. A series of experiments were conducted to compare LDA based on NENaN (NELDA) with representative filters, where the test sets of Dataset1&2 were consistent with Experiment I, while the train sets were randomly divided into labeled data (40%) and unlabeled ones. Moreover, we discussed the impact of update frequency on the classifier, where the classifier was initialed with 10% labeled data. Table II detailed the description and parameters of the state-of-art methods.

3)Experiment III investigated the performance of the presented filter in continuous multi-mental tasks (Dataset3).

TABLE II
COMPARISON ALGORITHMS AND PARAMETERS IN EXPERIMENT II

Symbol	Algorithm
SFCM	The classifier was self-training with FCM [19], and threshold $\varepsilon_1 = 1/C$, where C is the number of classes.
STDP	the structure of feature space was revealed by DPC [18]; pa=2.
STDPNF	the structure of feature space was revealed by DPC with a NaN based filter [29]; pa=2.
NoFNaN	NaN was used for structure discovering and NOF to filter out outliers [27].
PWKnn	Current test instance was assessed by a probabilistic weighted K-nearest neighbor (K=5) with a confidence threshold ($\Gamma = 0.75$)[10].
Pmean	The global mean and covariance matrix of LDA were updated without labels [35].
NENaN	After NaN revealed the data structure, a neighborhood editing method was used for filtering.

B. Experiment I: NENaN in Training Phase

We used a synthetic 2-dimensional dataset with 500 instances (half per class) to qualitatively illustrated the operation of NENaN. Fig.5 visually illustrates the outputs of NENaN: (a) is the initial distribution, (b) marks the mislabeled and isolated instances (total 46 instances) with circles, (c) marks the boundaries with circles, (d) picks out representatives of supporting instances by Algorithm 4, (e) depicts the distribution of 454 instances without interferences, (f) shows 162 core instances. As shown in Fig.5, NENaN removes mislabeled and isolated instances and retains boundaries and supporting instances that contributed significantly to robust the classification decision. Additionally, the large-scale supporting instances are pruned by the local representation method, which reduces the computation and storage consumption of self-training while preserving the shape of feature space faithfully.

Fig.6 depicts the average values of five times with and without NENaN in initialization. NENaN improves the LDA performance in all data sets, with an average accuracy of 79.5% and 76.74%, respectively. A one-sided signed-rank test result is 0.0348, proving the edited train set is beneficial for LDA initialization.

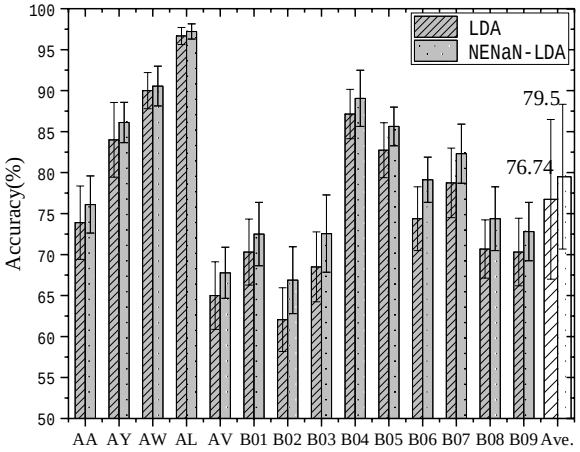


Fig.6. The accuracy of LDA with and without NENaN.

Since we got a better recognition result in the range of 0.01~0.1, we repeated the experiment at 0.01 step. An optimization η=0.03 was obtained, which would be the fixed update coefficient for subsequent studies.C. Experiment II: the merits of NENaN

C. Experiment II: The Merits of NENaN

1) compared with other filters

This section compared NENaN with exiting representative filters with LDA being basic classifier, and parameters referred to their work (see Table II).

Table III shows the recognition accuracy of different methods versus NENaN after five times repeats according to the setting in Experiment II. NENaN achieved good performance in general, STDPNF was also a competitive method, Pmean was the worst, with the average accuracies SFCM=71.76%, STDP=72.41%, NoFNaN=70.29%, PWKnn=75.42%, Pmean=66.63%, STDPNF=78.345%, NENaN=79.06%. A Wilcoxon signed-rank test illustrated that NENaN was significantly better than SFCM, STDP and NoFNaN, PWKnn, and Pmean (ρ-value<0.05), but had no significant difference relative to STDPNF (p-value= 0.4487). It comes down to the following reasons. One is that it is difficult to select suitable parameters for structural analysis in unstable feature space. Generally, the performance of parameter-dependent clustering methods such as SFCM, STDP, NoFNaN, and PWKnn is inferior to that of STDPNF, NENaN [28]. Pmean is nonparametric, but ignores the distribution difference of different classes [35]. Besides, PWKnn and STDP predict an unlabeled instance based on existing labeled data, and have poor performance in the insufficient situation. NoFNaN has the advantage of eliminating outliers, but it fails to handle mislabeled samples. By contrast, NENaN and STDPNF search natural neighbors by considering both labeled and unlabeled data, but STDPNF only focuses on mislabeled instances and ignores outliers that may induce a serious deviation in an adaptive classifier.

Fig.7 shows the training time on a data set consisting of 144 labeled instances and 216 unlabeled instances. The

PWKnn algorithm consumes the most time (3.741s). PWKnn time complexity is O(D*N*N), where D denotes the dimension and N is the number of samples. In light of time complexity, PWKnn is very unsuitable for high-dimensional data sets. SFCM and STDP consume relatively short time, but both need to calculate a distance matrix in advance, which the storage cost would be exceptionally high with the expansion of sample size. Compared with NoFNaN (O(Nlog(N))), NENaN spend shorter time (0.593s), which reflects the advantage of local representative in decreasing time consumption, and the benefit is even more outstanding in the long-running system. Although the classification performance difference between NENaN and STDPNF is not significant, NENaN is more competitive in time and storage complexity.

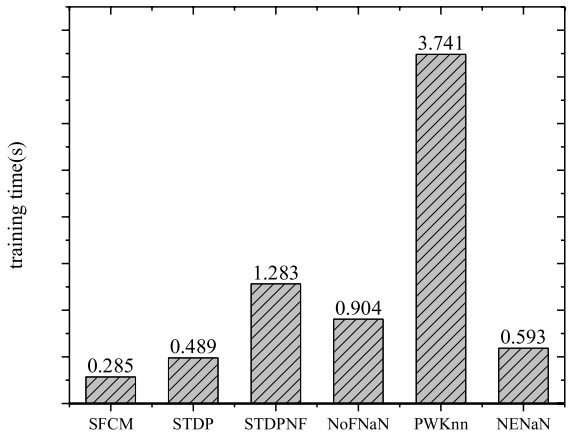


Fig.7. The running time of different filters.

2)the impact of update frequency on adaptive classifier

This section discussed the effect of different ratios of unlabeled data in each classifier update, i.e., update frequency on an adaptive classifier. The labeled train data is 10%, the rest for testing.Fig.8 depicts the accuracies of all algorithms on subjects 'B04' and 'B08'. The accuracies of all filters increase with the update frequency. When the update frequency is relatively high, the results do not increase accordingly. NENaN achieves better performance in nearly all scenarios. This may be that NENaN filters out the misclassifications and outliers, and incorporates high-confidence test data as train data, which mitigate the unsupervised clustering deviation caused by insufficient labeled samples.

To illustrate the advantages of NENaN in mining information of unlabeled instances, Fig.9 depicts the decision boundaries of STDP, STDPNF and our algorithm with three updates, where the number of labeled instances is 72, and 162 unlabeled ones each time. Fig.9 describes the adaptive process of the decision boundary as the test sample changes. Compared with STDPNF, STDP has a higher error rate. The reason is that although both STDP and STDPNF use DPC to explore the structure of unlabeled data, STDP has the risk of mislabeling, whereas STDPNF exploits NaN-based filter to delete misclassified instances. Compared with STDPNF, NENaN has smaller decision boundary bias and tracks newly added unlabeled samples better, which attributes to NENaN's ability to filter out misclassified samples and exclude instances far from the dataset center. The results indicate that our adaptive scheme can deal with feature drift effectively even in a small training set.

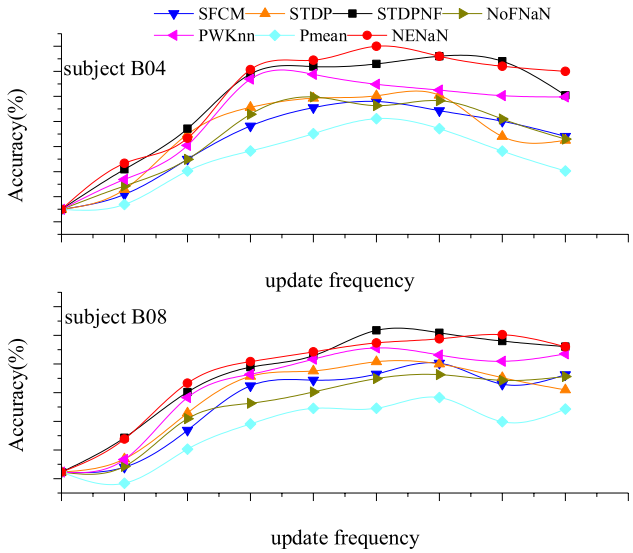
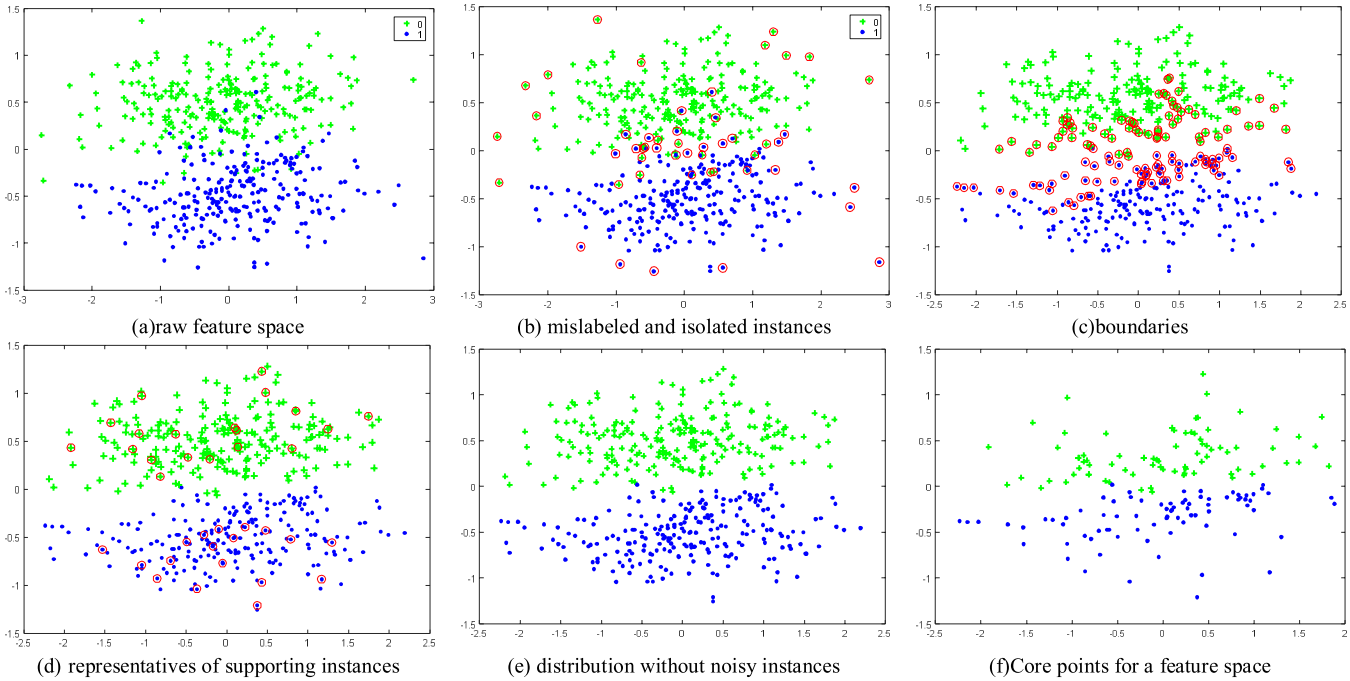


Fig.8. The performances in different update frequencies.



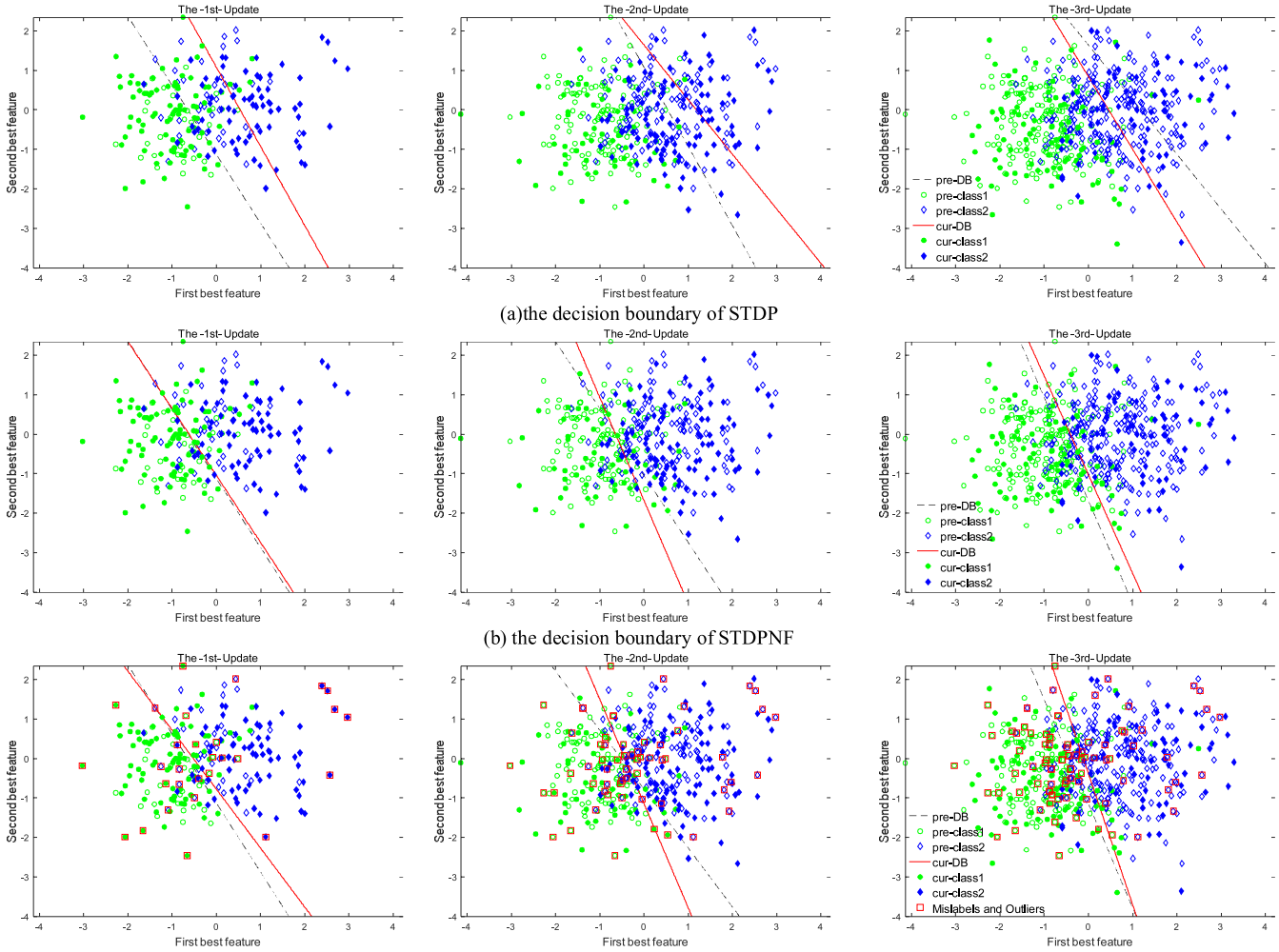
(a)raw feature space (b) mislabeled and isolated instances (c)boundaries

(d) representatives of supporting instances (e) distribution without noisy instances (f)Core points for a feature space

Fig.5. An example of NENaN dealing with a 2-dimensional synthetic dataset.

TABLE III
EXPERIMENT RESULTS (ACCURACY AND STD) OF DIFFERENT METHODS

	SFCM	STDP	STDPNF	NoFNaN	PWKnn	Pmean	NENaN
AA	66.67±2.54	69.29±3.12	75.71±2.49	66.78±3.31	74.44±3.49	61.9±5.16	**79.67±4.18**
AY	78.48±3.18	78.71±2.75	**88.01±3.06**	75.14±3.52	78.33±3.8	72.14±4.51	86.9±3.46
AW	78.03±4.05	76.38±3.37	83.3±3.15	71.67±3.45	79.76±4.25	69.05±3.79	**85.71±3.24**
AL	90.56±1.93	90.49±2.06	**96.67±1.41**	92.87±1.41	95.97±1.78	88.1±2.13	**96.67±1.04**
AV	59.52±3.15	60.71±2.86	67.56±3.14	62.22±3.73	65.36±3.61	55.96±5.13	**69.78±3.05**
B01	63.55±3.04	66.56±3.52	74.55±3.54	62.26±4.24	74.06±3.14	55.48±4.95	**75.63±2.73**
B02	54.31±2.49	54.58±4.15	**62.13±2.26**	53.89±4.31	54.27±4.15	53.49±4.48	60.71±3.17
B03	58.66±4.42	60.62±3.71	65.72±4.42	57.75±3.83	**67.5±3.53**	55.54±3.92	63.68±3.24
B04	83.75±3.35	82.64±4.06	90.43±3.63	81.89±3.74	87.38±3.39	78.57±3.64	**92.94±3.64**
B05	79.19±4.78	78.31±4.13	85.63±3.53	72.25±2.86	**86.69±3.04**	71.73±4.21	85.31±3.59
B06	74.67±3.49	76.25±3.45	**81.68±2.84**	74.72±2.78	79.42±2.43	72.46±4.27	80.79±2.81
B07	71.33±4.04	70.61±3.26	72.81±2.45	70.92±3.07	71.56±3.31	64.53±3.49	**73.82±3.27**
B08	73.25±3.54	76.52±3.17	**80.86±3.05**	70.13±3.85	77.81±3.29	67.29±5.08	79.39±3.41
B09	72.69±2.79	72.12±3.61	75.71±3.19	71.56±2.65	74.38±2.73	66.62±3.72	**80.86±3.11**
Average	71.76	72.41	78.63	70.29	76.21	66.63	**79.42**
$\rho - value$	**0.0156**	**0.0273**	0.4487	**0.0078**	**0.0408**	**0.0039**	--



(a)the decision boundary of STDP

(b) the decision boundary of STDPNF

(c) the decision boundary of NENaN

Fig.9. Adaptive decision boundaries of the subject 'B04' in three update frequency scenario. 'pre-DB' denotes 'previous decision boundary', and 'cur-DB'means 'current decision boundary'

D. Experiment III: Online Multiple Classifications

We extended our NENaN filter to a tri-categorical continuous EEG scenario with SVM, KNN(*k*=5), and LDA being the base classifier, respectively. We used the LibSVM toolbox with RBF kernel, and the five-fold cross validation to optimize the penalty parameter $c\,(2^{-8} \sim 2^{1})$ {} and kernel parameter σ $(2^{-8} \sim 2^{1})$. We set the first session for training and the rest three sessions for testing, and

re-estimated the classifier every 200 test trials. TABLE III depicts the performances of STDPNF, NENaN and the competition winner with Improved DB Discriminator (IDB). NENaN based classifier overwhelms the rivals in most scenarios, the average accuracies are 70.17%, 66.74%, 67.87%, respectively. Fig.10 shows the average time of eight times retraining. Although both NENaN and STDPNF filters are both based on the principle of natural neighborhood editing, NENaN consumes less time and has a smaller standard deviation. As a compromise between time consumption and accuracy, NENaN-based LDA is more applicable for continuous classification scenario.

TABLE III PERFORMANCE OVER THE THREE SUBJECTS.

Scenarios	STDPNF			NENaN			
	SVM	KNN	LDA	SVM	KNN	LDA	IDB
1→1	75.16	70.73	74.41	**77.36**	73.23	76.96	70.73
1→2	**80.06**	73.35	76.15	79.54	75.46	77.12	74.58
1→3	73.21	70.16	73.94	73.94	74.17	**77.45**	74.17
2→1	**69.48**	62.66	60.06	67.48	64.84	62.66	56.72
2→2	**76.25**	73.04	75.27	75.77	72.16	74.48	66.12
2→3	70.61	65.38	70.61	**72.32**	65.68	69.35	64.74
3→1	65.16	61.19	61.19	**67.16**	62.29	64.06	58.68
3→2	59.84	58.24	54.06	**60.04**	57.32	55.37	41.19
3→3	**60.45**	50.15	52.91	57.94	55.54	53.42	48.83
Mean	70.02	64.99	66.51	**70.17**	**66.74**	**67.87**	61.75
Std	6.29	6.93	8.45	6.77	6.64	8.44	10.26

'i→j' denotes the model is trained with the first session of the ith subject, and test with the jth session.

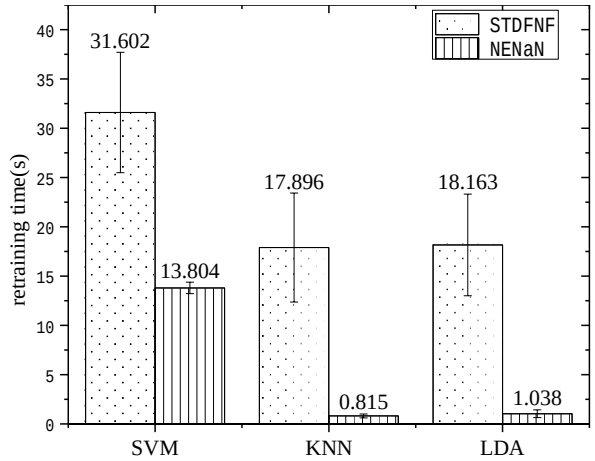


Fig.10. The average retraining time of each classifier

V. CONCLUSIONS AND FUTURE WORK

Aiming at the deficiency of train data and poor stability in the online MI-BCI system, we proposed an adaptive framework that integrates the test data into the self-training process to compensate for overfitting and mitigate distribution drifts. We design a nonparametric filter (NENaN) to evaluate the contribution of an instance to the classifier, filter out mislabeled samples and outliers, and retain the high-confidence data. Additionally, we introduce the local representative method to reduce the train set and produce a core set, which speeds up the retraining process and reduces the computation and storage consumption at the same time.

We conducted a series of experiments, where 5 representative filters and 3 datasets, a total of 17 subjects, were adopted. The experimental results demonstrate that: (a) NENaN improves the stability of LDA and has a better performance compared with the classifier without sample selection; (b) NENaN is a parameter-free filter and removes mislabeled samples and outliers by exploiting the underlying structure of unlabeled data, even when labeled data is insufficient; (c) NENaN based adaptive classifier is robust and performs pretty well in multi-task scenario;(d) The average running time is about 0.593s, which provides a promising method for an online BCI system. The subsequent work will focus on validating the real-time online MI-BCI systems with the proposed adaptive classifier in reality.

ACKNOWLEDGMENT

The authors would like to thank the Institute for Knowledge Discovery, Fraunhofer FIRST Intelligent Data Analysis Group, and IDIAP Research Institute for sharing their datasets, Qingsheng Zhu from Chongqing University for sharing the source code of STDPNF.

REFERENCES

[1] F. Lotte, L. Bougrain, A. Cichockiet, et al, "A Review of Classification Algorithms for EEG-based Brain-Computer Interfaces: A 10-year Update," *Journal of Neural Engineering, IOP Publishing*, vol.15, no.3, pp.1-56, 2018.

[2] Sang-Hoon Park, David Lee, and Sang-Goog Lee, "Filter Bank Regularized Common Spatial Pattern Ensemble for Small Sample Motor Imagery Classification," IEEE *Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 2, pp498-505, 2018.

[3] Fotis P. Kalaganis, Elisavet Chatzilari, and Spiros Nikolopoulos, et al, "An error-aware gaze-based keyboard by means of a hybrid BCI system," *Scientific Reports*, pp257-268, 2018.

[4] R. Chaisaen, P. Autthasan, N. Mingchinda, et al, "Decoding EEG Rhythms During Action Observation, Motor Imagery, and Execution for Standing and Sitting," *IEEE Sensors Journal*, vol.20, no.22, pp13776-13786, 2020.

[5] S. A. C. Yohanandan, I. Kiral-Kornek, J. Tang, B. S. Mshford, U. Asif and S. Harrer, "A Robust Low-Cost EEG Motor Imagery-Based Brain-Computer Interface," 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2018, pp5089-5092.

[6] Rui Zhang, Yuanqing Li, Yongyong Yan, et al, "Control of a Wheelchair in an Indoor Environment Based on a Brain–Computer Interface and Automated Navigation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol.24, no.1, pp128-139, 2016.

[7] Yu Zhang, Guoxu Zhou, Jing Jin, et al, "L1-Regularized Multiway Canonical Correlation Analysis for SSVEP-Based BCI," *IEEE Transactions on Neural Systems & Rehabilitation Engineering*, vol.21, no.6, pp887-896, 2013.

[8] Shiu Kumar, and Alok Sharma, "A new parameter tuning approach for enhanced motor imagery EEG signal classification," *Medical & Biological Engineering & Computing*, vol.56, no.10, pp1861-1874, 2018.

[9] Yong Jiao, Yu Zhang, Xun Chen, et al, "Sparse Group Representation Model for Motor Imagery EEG Classification," *IEEE Journal of Biomedical and Health Informatics*, vol.23, no.2, pp631-641, 2019.

[10] Haider Raza, Dheeraj. Rathee, Shang-Ming Zhou, et al, "Covariate Shift Estimation based Adaptive Ensemble Learning for Handling Non-Stationarity in Motor Imagery related EEG-based Brain-Com puter Interface," *Neurocomputing*, vol.343, no.28, pp154-166, 2019.

[11] Yu Zhang, Chang S. Nam, Guoxu Zhou, et al, "Temporally Const rained Sparse Group Spatial Patterns for Motor Imagery BCI," *IEE E Transactions on Cybernetics*, vol.49, no.9, pp3322-3332, 2019.

[12] Chen Minyou, Tan Xuemin, Zhang Li, "An iterative self-training support vector machine algorithm in brain-computer interfaces," *Intelligent Data Analysis*, vol.20, no.1, pp67-82, 2016.

[13] Jafar Tanha, Maarten Van Someren, Hamideh Afsarmanesh, "Semi-supervised self-training for decision tree classifiers," *International Journal of Machine Learning & Cybernetics*, nol.8, no.1, pp355-370, 2017.

[14] Joris Tavernier, Jaak Simm, Karl Meerbergen, et al, "Fast semi-supervised discriminant analysis for binary classification of large data sets," *Pattern Recognition*, vol.91, pp86-99, 2019.
[15] Wu Di, Shang Mingsheng, Luo Xin, et al, "Self-training semi-supervised classification based on density peaks of data," *Neurocomputing*, vol.275, no.31, pp180-191, 2018.
[16] Haitao Gan, Nong Sang, Rui Huang, et al, "Using clustering analysis to improve semi-supervised classification," *Neurocomputing*, vol.101, pp.290-298, 2013.
[17] Adrian Calma, Tobias Reitmaier, and Bernhard Sick, "Semi-Supervised Active Learning for Support Vector Machines: A Novel Approach that Exploits Structure Information in Data," *Information Sciences*, vol.456, pp.13-33, 2018.
[18] N. Piroonsup, and S. Sinthupinyo, "Analysis of training data using clustering to improve semi-supervised self-training," *Knowledge Based Systems*, vol.143, pp65-80, 2017.
[19] Caiwen Wang, and Youlong Yang, "Nearest Neighbor with Double Neighborhoods Algorithm for Imbalanced Classification, " IAENG International Journal of Applied Mathematics, vol.50, no.1, pp147-159, 2020.
[20] Ali Serhan Koyuncugil, and Nermin Ozgulbas, "Statistical Roots of Machine Learning, Deep Learning, Artificial Intelligence, Big Data Analytics and Data Mining," Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2019, 22-24 October, 2019, San Francisco, USA, pp320-322.
[21] Isaac Triguero, José A. Sáez, Julián Luengo, et al. "On the characterization of noise filters for self-training semi-supervised in nearest neighbor classification," *Neurocomputing*, vol.132, pp30-41, 2014.
[22] Ming Li, and Zhi-Hua Zhou, "SETRED: Self-training with Editing," Lecture Notes in Artificial Intelligence: Proceedings of the 9th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining, Springer-Verlag, May, 2005, Hanoi, VIETNAM, pp611-621.
[23] Qingsheng Zhu, Ji Feng, and Jinlong Huang, "Natural neighbor: A self-adaptive neighborhood method without parameter K," *Pattern Recognition Letters*, vol.80, pp30-36, 2016.
[24] Aiping Guo, Ajuan Jiang, and Zhangyu Cheng, "A Hybrid Clustering Method for Bridge Structure Health Monitoring," Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2018, 23-25 October, 2018, San Francisco, USA, pp161-165.
[25] Dongdong Cheng, Qingsheng Zhu, Jinlong Huang, et al. "Natural neighbor-based clustering algorithm with local representatives," *Knowledge Based Systems*, vol.123, pp238-253, 2017.
[26] Jinlong Huang, Qingsheng Zhu, Lijun Yang, and Ji Feng, "A non-parameter outlier detection algorithm based on Natural Neighbor," *Knowledge Based Systems*, vol.92, pp71-77, 2016.
[27] Lijun Yang, Qingsheng Zhu, Jinlong Huang, et al, "Natural Neighborhood Graph-based Instance Reduction Algorithm without Parameters," *Applied Soft Computing*, vol.70, pp279-287, 2018.
[28] Junnan Li, Qingsheng Zhu, and Quanwang Wu, "A self-training method based on density peaks and an extended parameter-free local noise filter for k nearest neighbor," *Knowledge-Based Systems*, vol.184, pp.1-12, 2019.
[29] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, et al, "LOF: Identifying Density-Based Local Outliers," Proceeding Paper in Acm Sigmod International Conference on Management of Data, *ACM*, May 16-18, 2000, DALLAS, TX, pp93-104.
[30] Wen Jin, Anthony K. H. Tung, Jiawei Han, and Wei Wang, "Ranking outliers using symmetric neighborhood relationship," Proceedings Paper in 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining, April 09-12, Singapore, SINGAPORE, 2006, pp577-593.
[31] Jihyun Ha, Seulgi Seok, and Jong-Seok Lee, "Robust outlier detection using the instability factor," *Knowledge-Based Systems*, vol.63, pp15-23, 2014.
[32] Clemens Brunner, Muhammad Naeem, Robert Leeb, Bernhard Graimann, Gert Pfurtscheller, "Spatial filtering and selection of optimized components in four class motor imagery EEG data using independent components analysis," *Pattern Recognition Letters*, vol.28, no.8, pp957-964, 2007.
[33] G. Dornhege, B. Blankertz, G. Curio and K. R. Muller, "Boosting bit rates in noninvasive EEG single-trial classifications by feature combination and multiclass paradigms," *IEEE Transactions on Biomedical Engineering*, vol.51, no.6, pp993-1002, 2004.
[34] J. R. Millan, "On the need for on-line learning in brain-computer interfaces," 2004 IEEE International Joint Conference on Neural Networks, July 25-29, 2004, pp2877-2882.
[35] C. Vidaurre, M. Kawanabe, P. von Buenau, et al, "Toward Unsupervised Adaptation of LDA for Brain–Computer Interfaces," *IEEE Transactions on Biomedical Engineering*, vol.58, no.3, pp587-597, 2011.

QIN JIANG received B.S. and M.S. in Biomedical Engineering from Chongqing University of Technology, China. She is currently pursuing a doctoral degree in Pattern recognition and knowledge discovery from Chongqing University of Posts and Telecommunications. Her research interests include machine learning, pattern recognition, biomedical signal process and brain-computer interface.

YI ZHANG received the Ph.D. degree in mechanical manufacturing and automation from Huazhong University of science and technology, Wuhan, China. He completed his post-doctoral training in intelligent multimode human-computer interaction at the University of Essex, London, UK. He is a professor and doctoral supervisor of Advanced Manufacturing Engineering College of Chongqing University of Posts and telecommunications. His research interests include robot automatic control and human-computer interaction.

GENGYU GE received the B.S. degree in information and computing science from Chuzhou University, Chuzhou, China, in 2011, and M.S. degree in computer system architecture from Southwest University, Chongqing, China, in 2014, respectively. He is currently a Ph.D. student at Chongqing University of Posts and Telecommunications. His research interests include mobile robot navigation, semantic slam and embedded system application.

WEI WANG received the M.S. degree in Mechanical & Electronic Engineering from Northwest Normal University, Lanzhou, China. He is currently a Ph.D. student at Chongqing University of Posts and Telecommunications. His research interests include Deep Learning and Pattern Recognition.