Incremental Learning Vector Quantization for Character Recognition with Local Style Consistency

Yuan-Yuan Shen[1,3] and Cheng-Lin Liu[1,2,3,4](✉)

[1] Research Center for Brain-Inspired Intelligence,
Institute of Automation of Chinese Academy of Sciences, Beijing 100190, China
[2] National Laboratory of Pattern Recognition,
Institute of Automation of Chinese Academy of Sciences, Beijing, China
liucl@nlpr.ia.ac.cn
[3] University of Chinese Academy of Sciences (UCAS), Beijing, China
[4] CAS Center for Excellence in Brain Science and Intelligence Technology,
CAS, Beijing, China

Abstract. Incremental learning is a way relevant to human learning that utilizes samples in online sequence. In the paper, we propose an incremental learning method called Incremental Adaptive Learning Vector Quantization (IALVQ) which aims at classifying characters appearing in an online sequence with style consistency in local time periods. Such local consistency is present commonly in document images, in that the characters in a paragraph or text line are printed in the same font or written by the same person. Our IALVQ method updates the prototypes (parameters of classifier) incrementally to adapt to drifted concepts globally while utilize the style consistency locally. For style adaptation, a style transfer mapping (STM) matrix is calculated on a batch of samples of assumed same style. The STM matrix can be used both in training for prototypes updating and in testing for labels prediction. We consider supervised incremental learning and active incremental learning. In the latter way, class labels are attached only to samples that are assigned low confidence by the classifier. In our experiments on handwritten digits in the NIST Special Database 19, we evaluated the classification performance of IALVQ in two scenarios, interleaved test-then-train and style-specific classification. The results show that utilizing local style consistency can improve the accuracies of both two test scenarios, and for both supervised and active incremental learning modes.

Keywords: Incremental learning · Style transfer mapping · Local style consistency · Incremental Adaptive Learning Vector Quantization

1 Introduction

The size of real-world database has increased dynamically, therefore it is necessary to update existing models based on continuously acquired data instead of

C.-L. Liu et al. (Eds.): BICS 2016, LNAI 10023, pp. 228–239, 2016.
DOI: 10.1007/978-3-319-49685-6_21

re-training them. Incremental learning is a branch of machine learning method centered on solving such problems. Classical incremental learning has been intensively studied in the machine learning research area [1–3]. The learning algorithm observes instances in a sequential manner (i.e., instances usually appear one-by-one, or batch-by-batch). In addition, incremental learning algorithms are always accompanied by distribution change (concept drift).

Various incremental learning algorithms have been developed for different pattern recognition applications. However, when it comes to incremental character recognition, there are some new characteristics to consider. First, character patterns are presented as groups frequently, which means we can utilize the local style consistency, i.e., patterns in a time period have the same style. For example, characters in a paragraph or a text line are usually printed in the same font or written by a same person. Second, the distribution change is ubiquitous. Characters from different sources (pages in different fonts or written by different persons) have different styles. Such style variation results in distribution change in the feature space. Last but not least, for the practical application of character recognition, we hope that the incremental learning can not only adapt to the changing styles in learning from sequence of patterns, but also remember all the styles in the previously learned patterns, because in application, the learned classifier will face test patterns of all possible styles.

To cope with the characteristics stated above in incremental character recognition, we propose incremental adaptive learning vector quantization (IALVQ). IALVQ can classify patterns appear in an online sequence with style consistency in local time periods. Learning vector quantization (LVQ) is a learning method for prototype classifier, which is akin to human learning in that prototypes (templates) are memorized in the brain to recognize new patterns and the prototypes can be learned by both generative and discriminative way. We use style transfer mapping (STM) [4] to compute the style matrix in consideration of local style consistency. After getting the style matrix, a pattern from feature space with changing style can be mapped into a style-free space. In order to adapt different styles, we bring in the forget mechanism in the way that old data are discarded or weakened to ensure that style transfer matrix is learned from the latest data. Prototype learning in style-free space is very helpful for the test-then-train scenario. However, we also hope the learned prototypes can memorize all the styles in the past data to enhance the performance of multi-style classification. To do this, we learn style-conscious prototypes as done in traditional Increment Learning Vector Quantization (ILVQ) simultaneously. In summarize, our proposed IALVQ is a new method which combines STM and ILVQ to realize incremental character learning and recognition.

The human learning of characters occasionally interacts with a teacher (i.e., inquire a teacher for the label when we face an un-known character). Inspired by this, we also introduce the active incremental learning mode of IALVQ. In this case, we attach labels to the training samples only when the confidence assigned by the classifier is low. Our experimental results show that in the case of active incremental learning, utilizing local style consistency by STM is also efficient to improve the classification performance.

The rest of this paper is organized as follows. Section 2 reviews related works. Section 3 presents the proposed IALVQ method in detail. Section 4 presents our experimental results and analysis. Section 5 is the conclusion of this paper, and a discussion of future directions is also provided.

2 Related Work

Our work is mainly related to previous works of incremental learning and style consistent learning. Some representative works are reviewed as follows.

Incremental learning has been extensively studied in recent years. Unlike traditional machine learning which assumes that all the training patterns are available before training, incremental learning is more relevant to practical environments where patterns appear in an sequential manner, and the pattern distribution may change over time. So, how to update classifier model (adapt pre-model into latest data) is the vital issue in incremental learning. One classical approach is the Perceptron algorithm [5] proposed in 1950s. Perceptron adopts a simple adaptive strategy for updating the weights of single-layer neural network when an incoming pattern is misclassified. Incremental prototype-based classifier [6,7] is another typical model which adapts newest data by updating two nearest prototypes from the genuine class and the rival class of input pattern. By using covariance matrix as confidence information of different dimensions, some second-order incremental algorithms [8,9] have been proposed. In [10] three guildlines about incremental learning for large-scale visual recognition were proposed. A comprehensive survey on incremental learning with concept drift can be found in [11].

In character learning and recognition, characters appearing together usually originates from the same source of consistent style. Such local consistency is commonly present in document images where the characters in a paragraph or text line are printed in the same font or written by the same person. Previous works have shown that exploiting style consistency can promote the performance of character recognition. Sarkar and Nagy [12] proposed an optimal style constrained classifier which processes entire fields of characters rendered in a consistent style. Veeramachaneni and Nagy [13,14] proposed a Gaussian quadratic discriminant filed classifier for field classification. In [15], Huang et al. learned a writer-specific LDA transformation matrix with the new labeled data in an incremental handwriting recognition. By learning a style transfer mapping (STM) matrix, Zhang and Liu [4] proposed a writer-specific adaptation method.

Although both incremental learning and style consistency have been actively studied, to the best of our knowledge, the combination of incremental learning and style consistency has been considered before. The closest methods to the work presented here are [15,16], which are to adapt an well-learned classifier to writer-specific data.

3 Proposed Methods

In this section, we first introduce the incremental adaptive LVQ, then extend to active incremental adaptive LVQ. At Last, we will outline two paradigms for evaluating the classification performance of incremental learning methods.

3.1 Incremental Adaptive Learning Vector Quantization

For M-class classification, prototype learning is to design a set of prototype vectors $m_{ij}(i = 1, 2, ..., M, j = 1, ..., n_i$. n_i is the number of prototypes in class i), usually by minimizing the empirical loss on a training set. An input pattern $x \in \mathbb{R}^d$ is classified to the class of the nearest prototype:

$$k = \arg\min_{i=1}^{M} \min_{j=1}^{n_i} \|x - m_{ij}\|_2^2 = G(x, m), \tag{1}$$

There are many variations of LVQ algorithm [17–19]. In this paper, we use the one of LOG-likelihood of Margin (LOGM) [19].

In our proposed method, we learn two types of prototypes for each class: one is style-conscious prototypes $m_{il_1}(i = 1, ..., M, l_1 = 1, 2, ..., L_1)$ which are completely the same with conventional ILVQ; the other is style-free prototypes $m_{il_2}(i = 1, ..., M, l_2 = 1, 2, ..., L_2)$, which are irrelevant to specific style. However, the learning of two types of prototypes is mutually independent.

For the learning of style-conscious prototypes, given that m_1 and m_2 are the nearest prototype to pattern x from the positive class and the one from the rival class, the posterior probability of x belonging to genuine class i (i.e., the probability of correct classification) can be approximated by the sigmoid function:

$$P(C_i|x) = \sigma(\xi(\|x - m_2\|^2 - \|x - m_1\|^2)), \tag{2}$$

where $\xi(\xi > 0)$ is a constant for tuning the smoothness of sigmoid function and the conditional log-likelihood loss of pattern x is $\phi(x) = -\log P(C_i|x)$.

As each pattern x is arriving in incremental setting, ILVQ updates two prototypes m_1 and m_2 by stochastic gradient descent [20]:

$$\begin{aligned} m_1 &= m_1 - \eta \frac{\partial \phi(x)}{\partial m_1}, \\ m_2 &= m_2 - \eta \frac{\partial \phi(x)}{\partial m_2}. \end{aligned} \tag{3}$$

The style-free prototypes are learned in a similar manner, as detailed in the following section.

In incremental setting, we assume that adjacent patterns share the same style. So, the style transfer matrix which maps pattern from style-conscious space to style-free space for current pattern can be computed from several latest past patterns approximately. The objective function of learning style transfer matrix is composed of three parts. The first part is from the past patterns, the second part is from the current patterns, and the last part is for regularization. So the objective function of STM is:

$$F(t) = DecayWeight * F(t-1) + \triangle F(t) + \beta \|A - I\|_F^2, \tag{4}$$

where $F(0) = 0$, $DecayWeight$ is the decay parameter, I is the identity matrix, the hyperparameter β controls the trade-off between style transfer and non-transfer. For every moment, in order to reduce the influence caused by the past patterns we decay the relative item.

Suppose that $X = \{x_i | i = 1, ..., b\}$ is a small batch of patterns arriving at time t, the second part is computed as:

$$\triangle F(t) = \sum_{i=1}^{b} \|Ax_i - \hat{t}\|_2^2, \tag{5}$$

where b is the batch size, A is style transfer matrix, x_i is a training pattern, $\hat{t}$ is the nearest style-free prototype from the genuine class from x_i.

Let

$$\begin{aligned} S(t) &= DecayWeight * S(t-1) + \triangle S, \\ s.t.\ \ \triangle S &= xx^T, S(0) = 0, \\ T(t) &= DecayWeight * T(t-1) + \triangle T, \\ s.t.\ \ \triangle T &= \hat{t}x^T, T(0) = 0, \end{aligned} \tag{6}$$

then the computation of A has a closed-form solution:

$$\begin{aligned} A &= QP^{-1}, \\ Q &= T(t) + \beta I, \\ P &= S(t) + \beta I. \end{aligned} \tag{7}$$

Please refer to [4] for more details.

With the mapping matrix A, we can map pattern x into a style-free space by:

$$\hat{x} = Ax. \tag{8}$$

And then, we update the style-free prototypes as follows similar to Eq. (3):

$$\begin{aligned} m_1 &= m_1 - \eta \frac{\partial \phi(\hat{x})}{\partial m_1}, \\ m_2 &= m_2 - \eta \frac{\partial \phi(\hat{x})}{\partial m_2}. \end{aligned} \tag{9}$$

We summarize the process of learning style-free prototypes in Algorithm 1.

3.2 Active Incremental Adaptive Learning Vector Quantization

In character recognition, active incremental learning is very helpful. By partial interaction between the learner and the environment we can label for patterns

Algorithm 1. Learning of style-free prototypes

Input: Prototypes m_{il_2}
Output: Prototypes m_{il_2}

Initial style matrix $A = I$
while Receive new patterns X **do**
  Compute style-free patterns for each pattern in X by Eq. (8)
  Update style-free prototypes m_{il_2} by Eq. (9)
  Compute new style matrix A by Eq. (7)
end while
return m_{il_2}

that are assigned low confidence by the classifier. Due to the important effect of patterns that are assigned low confidence, active learning can boost the performance of classifier by requiring rare patterns.

In active incremental learning, how to evaluate the confidence for coming pattern is a vital problem. We use a simple but effective strategy that the confidence f for pattern x is calculated by Eq. (2) (posterior probability). Because of the unknown information of genuine class, we compute two nearest prototypes from the top two classes instead of the positive class and the rival class. Only when the confidence f is smaller than a predefined threshold p, the input pattern requests a label. Obviously, the larger the predefined threshold p is, the more are the required labels. For pattern x the conditional log-likelihood loss in active ILVQ and active IALVQ is computed as $\phi(x) = -f \log P(C_i|x)$. The active incremental adaptive learning vector quantization is summarized as Algorithm 2.

Algorithm 2. Active Incremental Adaptive Learning Vector Quantization

Input: Prototypes m_{ij}, Predefined threshold p
Output: Prototypes m_{ij}

initial style matrix $A = I$
while receive new patterns X **do**
  for each pattern x in X **do**
    Compute confidence f by Eq. (2)
    Decide whether to query the label $(Z = 1)$ or not $(Z = 0)$
    if $Z = 1$ **then**
      Query label y
      Set confidence $f = 1$
    end if
    Update style-conscious prototypes m_{il_1} using pattern x by Eq. (3)
    Compute style-free pattern $\hat{x}$ by Eq. (8)
    Update style-free prototypes m_{il_2} using style-free pattern $\hat{x}$ by Eq. (9)
  end for
  Compute new style matrix A by Eq. (7)
end while
return m_{ij}

3.3 Evaluation

We evaluate our proposed algorithm in two scenarios: interleaved test-then-train and style-specific classification.

Interleaved Test-Then-Train. Each pattern is used to test the classifier before it's used for training, so the classifier is always being tested on patterns it has not seen before. This evaluation method can make maximum use of the available data.

Style-Specific Classification. After training an incremental classifier, we test the generalization performance of the classifier. During test time, a batch of b patterns with the same style are given once. The process of testing is reported in Algorithm 3. Note that we use style-conscious prototypes for first-round classification for obtaining the initial labels of patterns. Then in style adaptive classification, style-free prototypes are used because the patterns are mapped toward style-free space.

Algorithm 3. Style-Specific Classification

Input: Prototypes m_{ij}
Style-specific unlabeled data $\{x_k\}_{k=1}^{b}$
Output: Predicted labels $\hat{y}_k, k = 1...b$
Initial style matrix $A = I$
for k=1:b **do**
  Compute label $\overline{y}_k = G(x_k, m_{il_1})$ using Eq. (1)
  Find nearest prototype $d_k = \arg\min_{l_2=1}^{L_2} \|x_k - m_{\overline{y}_k l_2}\|_2^2$
end for
for iter = 1:iterNum **do**
  Learn matrix style A using Eq. (7)
  for k=1:b **do**
    Predict label $\hat{y}_k = G(Ax_k, m_{il_2})$ using Eq. (1)
    Find nearest prototype $\hat{t}_k = \arg\min_{l_2=1}^{L_2} \|Ax_k - m_{\hat{y}_k l_2}\|_2^2$
  end for
end for

4 Experiments and Results

We evaluate the performance of the proposed IALVQ method on NIST handwritten digit data. Two learning modes were considered:

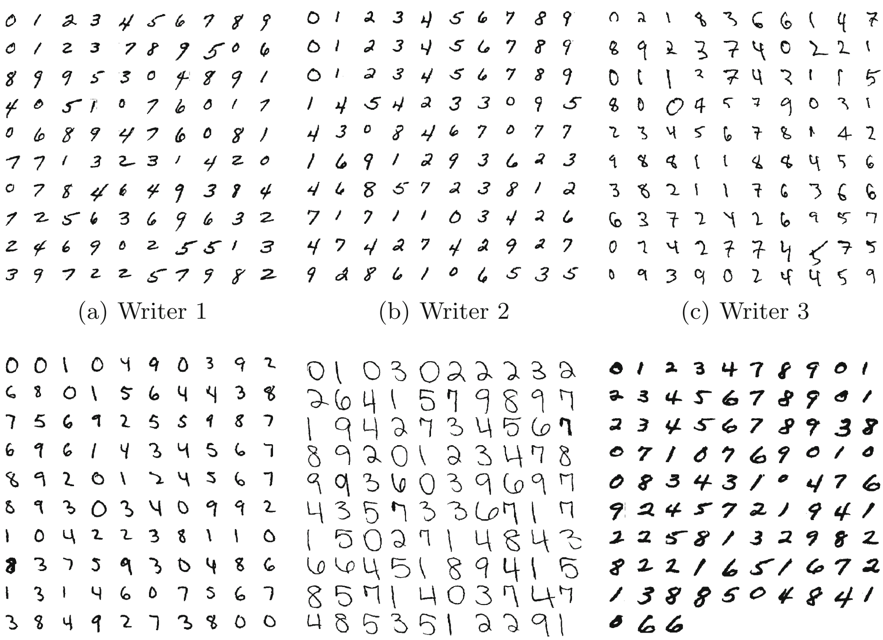
(i) supervised incremental learning;
(ii) active incremental learning.

Table 1. Handwritten numeral datasets

	Writers	Number of samples
SD3-Train	No.0-No.399 (400)	42969
SD7-Train	No.2100-No.2199 (100)	11585
SD3-Test	No.400-No.799 (399)	42821
SD7-Test	No.2200-No.2299 (100)	11660

4.1 Database

To test our proposed method on realistic data, we experimented with the datasets SD3 and SD7, which are contained in the NIST Special Database SD19 [21]. The datasets contain patterns of handwritten numerals labeled by writer and class. From SD3, we use samples of 400 writers for training and 399 writers for testing, and from SD7, samples of 100 writers for training and 100 writers for testing. The patterns of each writer are assumed to have the same writing style. The statistics of patterns for SD3 and SD7 are listed in Table 1. The patterns from some writers are shown in Fig. 1. Our choice of data is similar to that of [13] for adaptive classification.



(a) Writer 1 (b) Writer 2 (c) Writer 3

(d) Writer 4 (e) Writer 5 (f) Writer 6

Fig. 1. Samples of handwritten digits from 6 different writers.

4.2 Experimental Setting

All these patterns are arranged by writer order and patterns are permuted randomly within one writer. We extracted 100 blurred directional (chain-code) features from each pattern [22]. A small number of patterns are used to construct the initial prototypes. After getting initial classifier, new coming patterns are used for both prototypes updating and estimating style transfer matrix. The classification performance are evaluated using interleaved test-then-train and style-specific classification.

For all the datasets, we use very small portion patterns (200 patterns) for initialization. The trade-off parameter β in style adaptation is set as in [4]: $\beta = \hat{\beta}\frac{1}{d}Tr(\sum_{i=1}^{n} x_i x_i^T)$, where n is the total number of patterns that have been seen in model and $\hat{\beta}$ is set as 3 in our experiments. The smoothing parameter ξ is initialized as $2/cov$ like [19], where cov is the average covariance estimated from training data. The initial rate of gradient descent is set as 1 and the learning rate of the n-th pattern is calculated by adagrad algorithm [23]. The batch size b is set as 5 and $delayParam$ is set as 0.9.

4.3 Results and Discussions

Interleaved Test-Then-Train. Interleaved test-then-train is most frequently-used in the evaluation of incremental learning. By testing pattern before being used for training, we can make maximum use of patterns. The result assessed by this method can better reflect the instant performance of model. We list the results of ILVQ and IALVQ in Table 2 and Table 3 for supervised and active mode, respectively, in interleaved test-then-train.

Table 2. Error rates on four datasets using supervised ILVQ and IALVQ in interleaved test-then-train.

Dataset	Incremental LVQ	IALVQ
SD3-Train	1.66 %	1.31 %
SD7-Train	4.17 %	3.45 %
SD3-Test	1.37 %	1.25 %
SD7-Test	3.68 %	3.25 %

When considering local style consistency, we combine STM with incremental LVQ. For fair comparison, we use 3 prototypes for every class in both ILVQ and IALVQ. From the results in Table 2, we can see that compared to the conventional incremental LVQ, IALVQ can effectively reduce the error rate of interleaved test-then-train, by utilizing the local style consistency of writer-specific samples. This can be attributed to better class separability resulted in via feature transformation from STM. From the results in Table 3, we can see that due to the availability of only small proportion of samples, active ILVQ yields higher

Table 3. Error rates on four datasets using active ILVQ and IALVQ in interleaved test-then-train.

Dataset	Active ILVQ	# of used patterns	Active IALVQ	# of used patterns
SD3-Train	1.93 %	1497/42969	1.77 %	1366/42969
SD7-Train	4.38 %	1043/11585	4.17 %	942/11585
SD3-Test	1.67 %	1571/42821	1.46 %	1391/42821
SD7-Test	3.90 %	1133/11660	3.63 %	981/11660

error rate than supervised ILVQ (in Table 2). Active IALVQ also yields higher error rate than the supervised IALVQ. However, compared to active ILVQ, active IALVQ results in reduced error rate due to the effect of local style consistency utilization via STM.

Style-Specific Classification. In order to evaluate the generalization performance of our proposed method, we compare 4 different supervised methods and 2 different active methods with our method. Tables 4 and 5 show the classification error rates for supervised and active mode, respectively. A/B represents that A is adopted in training stage and B is adopted in testing stage. In testing stage, NN is classified based on the nearest prototype without considering style consistency, while STM utilizes style consistency of writer-specific data. For fair comparison, we use the best prototypes parameters respectively, that is, 5 prototypes are used in ILVQ while 5 style-conscious prototypes and 2 style-free prototypes are used in IALVQ.

Table 4. Error rates using supervised ILVQ and IALVQ in style-specific classification.

Training set	Test set	LVQ/NN	LVQ/STM	ILVQ/NN	ILVQ/STM	IALVQ/STM
SD3-Train	SD3-Test	1.02 %	0.79 %	1.58 %	1.12 %	0.99 %
SD3-Train	SD7-Test	4.19 %	3.21 %	5.96 %	4.49 %	4.26 %
SD7-Train	SD3-Test	1.79 %	1.30 %	2.70 %	2.01 %	1.80 %
SD7-Train	SD7-Test	2.34 %	1.87 %	3.55 %	2.82 %	2.51 %

In the active incremental learning setting, the number of labeled patterns in SD3-Train is 1441 for ILVQ (1556 for IALVQ), which accounts for about only 3 % of total data, and 1114 for ILVQ (1001 for IALVQ) in SD7-Train, which accounts for about 10 % of total data. The results in Table 4 show that when evaluating generalized classification performance using NN without considering style consistency, the incremental LVQ results in higher error rate than supervised LVQ which treats all training samples iteratively. Style-specific classification by STM reduces the error rate of both LVQ and ILVQ considerably.

Table 5. Error rates using active ILVQ and IALVQ in style-specific classification.

Training set	Test set	Active-ILVQ/NN	Active-ILVQ/STM	Active-(IALVQ)/STM
SD3-Train	SD3-Test	1.52 %	1.13 %	0.99 %
SD3-Train	SD7-Test	5.87 %	4.67 %	4.65 %
SD7-Train	SD3-Test	3.90 %	2.60 %	2.34 %
SD7-Train	SD7-Test	4.09 %	3.15 %	2.86 %

This observation conforms with previous results in [4]. When comparing ILVQ/STM and IALVQ/STM, we can see that the proposed IALVQ method can further reduce the error rate of ILVQ even when style consistency is considered in testing. The results in Table 5 confirms that style-specific classification with STM can also reduces the error rate of active ILVQ. Comparing active IALVQ with active ILVQ, again the proposed active IALVQ can reduce the error rate of active ILVQ when style consistency is considered in testing.

5 Conclusion

In this paper, we propose an incremental learning method utilizing local style consistency of samples for improving the classification performance of character recognition. Experimental results have shown that the proposed method is effective in both supervised incremental learning and active incremental learning. Active incremental learning is akin to human learning which occasionally interacts with a teacher for inquiring labels for unknown patterns. In our experiments, however, the proportion of inquired samples is considerable and implies high cost of interactive learning. In the future, we will seek to realize efficient unsupervised incremental learning or interactive learning with very small number of samples inquired.

Acknowledgments. This work has been supported in part by the National Basic Research Program of China (973 Program) under Grant 2012CB316302 and the Strategic Priority Research Program of the CAS under Grant XDB02060009.

References

1. Syed, N., Liu, H., Sung, K.: Incremental learning with support vector machines. In: International Joint Conference on Artificial Intelligence, pp. 352–356. Morgan Kaufmann Publishers, Sweden (1999)
2. Hoi, S.C.H., Wang, J., Zhao, P.: LIBOL: a library for online learning algorithms. J. Mach. Learn. Res. **15**(1), 495–499 (2014)
3. Zhao, P., Hoi, S.C.: OTL: a framework of online transfer learning. In: Proceedings of the 27th International Conference on Machine Learning, pp. 1231–1238 (2010)
4. Zhang, X.Y., Liu, C.L.: Writer adaptation with style transfer mapping. IEEE Trans. Pattern Anal. Mach. Intell. **35**(7), 1773–1787 (2013)

5. Rosenblatt, F.: The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev. **65**(6), 386–408 (1958)
6. Kirstein, S., Wersing, H., Körner, E.: A biologically motivated visual memory architecture for online learning of objects. Neural Netw. **21**(1), 65–77 (2008)
7. Xu, Y., Shen, F., Zhao, J.: An incremental learning vector quantization algorithm for pattern classification. Neural Comput. Appl. **21**(6), 1205–1215 (2012)
8. Cesa-Bianchi, N., Conconi, A., Gentile, C.: A second-order perceptron algorithm. SIAM J. Comput. **34**(3), 640–668 (2005)
9. Crammer, K., Dredze, M., Kulesza, A.: Multi-class confidence weighted algorithms. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 496–504 (2009)
10. Ushiku, Y., Hidaka, M., Harada, T.: Three guidelines of online learning for large-scale visual recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3574–3581 (2014)
11. Gama, J., ŽliobaitėI, B.A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Comput. Surv. (CSUR) **46**(4) (2014)
12. Sarkar, P., Nagy, G.: Style consistent classification of isogenous patterns. IEEE Trans. Pattern Anal. Mach. Intell. **27**(1), 88–98 (2005)
13. Veeramachaneni, S., Nagy, G.: Adaptive classifiers for multisource OCR. Doc. Anal. Recognit. **6**(3), 154–166 (2003)
14. Veeramachaneni, S., Nagy, G.: Style context with second-order statistics. IEEE Trans. Pattern Anal. Mach. Intell. **27**(1), 14–22 (2005)
15. Huang, Z., Ding, K., Jin, L., et al.: Writer adaptive online handwriting recognition using incremental linear discriminant analysis. In: Proceedings of the Conference on Document Analysis and Recognition, pp. 91–95. IEEE (2009)
16. Ding, K., Jin, L.: Incremental MQDF learning for writer adaptive handwriting recognition. In: Proceedings of the Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 559–564 (2010)
17. Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J., Torkkola, K.: LVQ PAK: the learning vector quantization program package. Technical report, Helsinki University of Technology (1995)
18. Kuncheva, L.I., Bezdek, J.C.: Nearest prototype classification: clustering, genetic algorithms, or random search? IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) **28**(1), 160–164 (1998)
19. Jin, X.B., Liu, C.L., Hou, X.: Regularized margin-based conditional log-likelihood loss for prototype learning. Pattern Recognit. **43**(7), 2428–2438 (2010)
20. Bishop, C.M.: Pattern recognition and Machine Learning. Springer, New York (2006)
21. Grother, P.: Handprinted forms and character database, NIST special database 19. Technical report and CDROM (1995)
22. Liu, C.L., Sako, H., Fujisawa, H.: Performance evaluation of pattern classifiers for handwritten character recognition. Int. J. Doc. Anal. Recognit. **4**(3), 191–204 (2002)
23. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**, 2121–2159 (2011)