# A Framework for Adaptation of the Active-DTW Classifier for Online Handwritten Character Recognition

Vandana Roy, Sriganesh Madhvanath, Anand S., Raghunath R. Sharma

HP Laboratories
HPL-2009-329

**Keyword(s):**
online handwritten character recognition, classifier, adaptation, Active-DTW

**Abstract:**

Practical applications of online handwritten character recognition demand robust and highly accurate recognition along with low memory requirements. The Active-DTW [11] classifier proposed by Sridhar et al. combines the advantages of generative and discriminative classifiers to address the similarity of between-class samples, while taking into account the variability of writing styles within the same character class. Active-DTW uses Active Shape Models to model the significant writing styles in a memory efficient manner. However, in order to create accurate models, a large number of training samples is needed up front, which is not desirable or available in many practical applications. In this paper, we propose a supervised adaptation framework for the Active-DTW classifier which allows recognition to begin with a small number of training samples, and adapts the classifier to the new samples presented to the system during recognition. We compare the performance of Active-DTW using the proposed adaptation framework, with a nearest-neighbor classifier using an LVQ-based adaptation scheme, on the online handwritten Tamil character dataset.

External Posting Date: September 21, 2009 [Fulltext] Approved for External Publication
Internal Posting Date: September 21, 2009 [Fulltext]

Published in International Conference on Document Analysis and Recognition, Barcelona, August 2009.

# A Framework for Adaptation of the Active-DTW Classifier for Online Handwritten Character Recognition

Vandana Roy and Sriganesh Madhvanath
Hewlett Packard Labs
Bangalore, India
{vandana.roy,srig}@hp.com

Anand S and Raghunath R. Sharma
Sri Satya Sai University
Puttaparthi, India
raghu.dmacs.psn@sssu.edu.in

## Abstract

*Practical applications of online handwritten character recognition demand robust and highly accurate recognition along with low memory requirements. The Active-DTW [11] classifier proposed by Sridhar et al. combines the advantages of generative and discriminative classifiers to address the similarity of between-class samples, while taking into account the variability of writing styles within the same character class. Active-DTW uses Active Shape Models to model the significant writing styles in a memory-efficient manner. However, in order to create accurate models, a large number of training samples is needed up front, which is not desirable or available in many practical applications. In this paper, we propose a supervised adaptation framework for the Active-DTW classifier which allows recognition to begin with a small number of training samples, and adapts the classifier to the new samples presented to the system during recognition. We compare the performance of Active-DTW using the proposed adaptation framework, with a nearest-neighbor classifier using an LVQ-based adaptation scheme, on the online handwritten Tamil character dataset.*

## 1. Introduction

Online handwritten character recognition is faced with new challenges due to the growing need for supporting pen-based text input methods on various mobile computers such as modern mobile phones and Smartphones, in a variety of languages and scripts. These devices are constrained by relatively low computational power and limited memory. Many of the scripts being addressed, such as the Indic scripts, consist of a large number of different characters with similar shapes, and different writing styles for the same character.

The Active-DTW classifier proposed by Sridhar *et al* [11] combines advantages of generative and discriminative classifiers to address these issues. Active-DTW uses Active Shape Models to capture distinct writing styles of each character in a compact form, and incorporates the discriminating content of the samples that do not naturally form clusters. However, in order to create accurate models, a large number of training samples is needed, which are often unavailable and expensive to collect for new scripts.

In this paper, we propose a supervised adaptation framework for the Active-DTW classifier to address these challenges. The adaptation framework allows the recognizer to be deployed using a very small training set, and adapt the models using samples of new styles over time. The paper is organized as follows. We first briefly describe the Active-DTW classifier and review related literature in Active Shape Models and writer adaptation for online handwritten character recognition. The adaptation framework for Active-DTW is then described. We then experimentally evaluate the performance of Active-DTW with adaptation, and compare it with a nearest-neighbor classifier using an LVQ-based adaptation scheme, on the online handwritten Tamil character dataset. We conclude the paper with a discussion of future research directions.

## 2. Active-DTW Classifier

The Active-DTW classifier is based on using Active Shape Models to capture writing styles of each character class. Distinct writing styles are determined by clustering the training samples. Active Shape Models [3] are the statistical models of a set of samples, that capture the principal variations of samples from the mean of the set. The principal variations are represented by a set of orthogonal eigen vectors obtained by principal component analysis (PCA). A valid deformation of a model is a weighted sum of the principal variations to the mean, within the limits described by the model. Assuming a Gaussian distribution, the data projected on the principal eigenvectors form a hyper-ellipsoid, whose boundaries are given by the corresponding eigen values.Thus, new sample can be generated from the model

by applying the limits on the boundary of hyper-ellipsoid which is generally order of thrice the square root of eigenvalues.

Training samples that do not form prominent clusters typically correspond to rarely occurring writing styles within the training set. Active-DTW stores these as individual "free samples".

The objective of Active-DTW is to compute an optimal deformation sample of a model that minimizes the distance to the test sample. Active-DTW works by classifying the test sample as the class to which the distance is minimum. Distance to a class is defined as the minimum distance computed using Dynamic Time Warping (DTW) of the test sample to all the optimal deformations and free samples of the class.

### 2.1. Training and Recognition

Training the Active-DTW classifier requires clustering the training dataset, and creating shape models for each character, as in Algorithm-1.

**Algorithm 1** Active-DTW Classifier: Training

1: $V_{nc}$, $E_{nc}$, $\mu_{nc}$, $M_{nc}$: Eigen vectors, eigen values, mean, number of samples of $n^{th}$ cluster of class $c$
2: **for** each class $c = 1 \ldots C$, **do**
3: Cluster the training samples of class $c$
4: **for** each cluster $n_c = 1 \ldots N_c$ & $M_{nc} > MinClusterSize$, **do**
5: Perform eigen analysis of cluster $n_c$, compute $\{V_{nc}, E_{nc}, M_{nc}, \mu_{nc}\}$
6: **end for**
7: **for** each cluster $n_c = 1 \ldots N_c \& M_{nc} \leq MinClusterSize$ **do**
8: Add the samples in cluster $n_c$ to the set $S_{nc}$ of free samples of class $c$
9: **end for**
10: Represent the shape model of class $c$ as $\Theta_c = \{V_{nc}, E_{nc}, M_{nc}, \mu_{nc}, S_{nc}\}$
11: **end for**

For our experiments, $MinClusterSize$ was determined empirically and set to 2.

In order to classify a given test sample, the optimal deformation from each Active Shape Model is computed. The optimal deformation corresponds to the nearest sample that can be generated using the model as described in [11]. The DTW distance of the test sample to each of the optimal deformations and the free samples of every class is computed. The test sample is assigned label of the class to which the DTW distance computed is minimum, as in Algorithm-2.

**Algorithm 2** Active-DTW Classifier: Recognition

1: $T$: The test sample
2: Return value: $c$: Label of the class
3: **for** each class $c = 1 \ldots C$ **do**
4: **for** each cluster $n_c = 1 \ldots N_c$ **do**
5: Compute the optimal deformation sample, $D_{nc}$
6: **end for**
7: Compute the minimum DTW distance from $T$ to all the optimal deformations $D_c$, and free samples $S_c$ as $Active - DTWdistance(T, c) = min\{DTWdistance(T, D_c), DTWdistance(T, S_c)\}$
8: Return label of the class to which the DTW distance is minimum $c = argmin_{c=1\ldots C}(Active{-}DTWdistance(T, c))$
9: **end for**

## 3. Background and Related Work

Active Shape Model (ASM) [3] is a technique similar to the Active Contour Model [8] to model variations in a class, but has the advantage that instances of an ASM can only deform in the ways found in its training set. A detailed description of training and applications of ASM is presented by Cootes *et al.* [4]. Training the ASM requires representing shape of an object using a number of landmark points. The within-shape variations are represented using PCA. ASM has widely been used in the pattern recognition problems involving large between-class variability. They have profound applications in facial features extraction [15], detecting variable objects, like hands and face [6] and for medical image analysis [5].

There has been some work in the area of writer adaptation for online handwritten character recognition. Szummer *et al.* [12] proposed a method to adapt a writer-independent classifier to individual writers using the method of experts. Adaptation to a writer is carried out by choosing a combination of classifiers for that user. Vouri *et al.* [14, 13] proposed various strategies for online adaptation of nearest-neighbor classifiers for online handwritten character recognition, using Learning Vector Quantization (LVQ) [9]. The basic strategies included adding new prototypes, reshaping existing prototypes, inactivating confusing prototypes, and combinations of these basic strategies. For example, the adaptation strategy *Add* adds the input character into the prototype set if the closest prototype belongs to a wrong class. *AddAndLvq*-strategy reshapes the nearest prototype if it belongs to the correct class. Otherwise, the input character is added to the prototype set. The reshaping of the prototypes is based on a modified version of the LVQ [10]. In addition, they propose to control these adaptation strategies by setting an upper limit on the number of prototypes or switching the adaptation on or off, depending on the classifier's performance.

## 4. An Adaptation Framework for Active-DTW

We propose a framework to adapt the shape models created by Active-DTW classifier during training, using new labeled samples encountered during testing. Incrementally adapting a model of class $c$ to a new sample requires computation of the modified model $\Theta_c^{new}$ from the already learned model $\Theta_c$ and the test sample $T_{new}$.

Based on whether the test sample is closer to model of a cluster or to a free sample of the class, either the model corresponding to the cluster or the set of free samples is modified. The following two subsections describe the method to adapt models of clusters and free samples.

### 4.1. Adapting model of a cluster

We derive the new ASMs of each cluster along the lines of method described by Hall *et al* [7]. This involves incremental computation of the eigenvalues, eigenvectors, mean of the samples, and number of samples in the cluster. Adapting a cluster $N_c$ results in the new values $E_{nc}^{new}, V_{nc}^{new}, \mu_{nc}^{new}, M_{nc}^{new}$ using $V_{nc}, E_{nc}, M_{nc}, \mu_{nc}$ and the test sample $T_{new}$. As the new values are directly computed from the previously computed values, there is no space overhead to store all the training samples, and the covariance matrices.

### 4.2. Adapting set of free samples

Adapting the set of free samples of a class involves adding the new sample to the set, and clustering these samples, if needed. The following steps outline the process of adapting the set of free samples of a class.

1. Add the new sample $T_{new}$ to the set $S_c$ of free samples of the class $c$.
2. If the number of samples in $S_c$ exceeds a threshold, cluster samples in $S_c$. This results in a new set of clusters and free samples of class $c$. Create the ASM of the new clusters as described in Algorithm-1.

If the number of free samples in a set exceeds a given threshold, they are clustered to ensure creation of models if sufficient number of samples of a rarely occurring style is encountered. The threshold value was determined empirically and set to 3 times the value of $MinClusterSize$. Adapting the set of free samples results in a modified set of free samples and/or increased number clusters of the class.

The adaptation framework follows different strategies for correct and incorrect recognition of test samples.

### 4.3. Adaptation in event of correct recognition

Adaptation is needed even if the decision made by classifier was correct. This is to ensure that when the system starts with a small training set, each cluster gets sufficient number of samples over time to construct accurate models. In the event of correct recognition, the following steps are carried out:

1. Identify whether the test sample was closer to a cluster or to a free sample of the class.
2. If the sample was closer to a cluster, and the number of samples in the cluster was less than a threshold, adapt the cluster as described in Section-4.1. This is to ensure that model does not over-fit with large number of training samples. The threshold value was determined empirically and set to 10.
3. If the sample was closer to a free sample, adapt the set of free samples, as described in Section-4.2.

### 4.4. Adaptation in event of incorrect recognition

When decision made by the recognizer is incorrect, the model corresponding to the true class of the test sample is modified. The incorrect recognition could be because of a totally new style of the true class being encountered for recognition, or a poor model of one of the existing styles. Following steps are carried out in order to adapt shape model in the event of incorrect recognition.

1. Identify the true class of the mis-recognized sample.
2. Identify whether the sample was closer to a model of one of the clusters of the class, or to a free sample.
3. If the sample was closer to a model, always adapt the model as described in Section-4.1.
4. If the sample was closer to a free sample, adapt the set of free samples, as described in Section-4.2.

## 5. Experiments and Results

In order to evaluate the performance of Active-DTW using the proposed adaptation framework, we used the HP Labs Isolated Handwritten Tamil Character Dataset [2]. The dataset consists of approximately 500 isolated samples each of 156 Tamil characters written by native Tamil writers.

**Preprocessing and Feature Extraction** Lipi Toolkit [1] was used for preprocessing and feature extraction. The character samples were size-normalized and resampled to 60 equidistant points. Then, point-based features were extracted from the preprocessed samples using the PointFloat feature extractor in Lipi Toolkit. The features at each point consisted of normalized coordinate values and the angles made at each point with the previous and next point in the stroke.

**Training strategy** Agglomerative hierarchical clustering from Lipi Toolkit was used to cluster the training samples of each class into a number of clusters determined by the L-method. In order to train the Active-DTW classifier, a variance of 4 times the eigenvalues was used to generate the allowed deformation samples.

**Evaluation strategy** The performance of adaptation was computed as average accuracies over various bins, where each bin consists of a specified number of test samples. Accuracy in each bin was computed as the average accuracy on samples of the bin and a specified number of samples (overlap) from the previous bin. The final accuracy was computed over a specified number of samples (final bin).

**Experiment 1: Performance of Active-DTW with adaptation** In order to simulate a realistic application scenario, 6 samples selected randomly per class were used as initial training samples and 20 samples per class were used for recognition and adaptation. The samples were presented randomly across classes during testing and the accuracies were computed over bins of 150 samples, with an overlap of 30 samples across bins.

It can be observed from Figure-1 that while average accuracy of the basic Active-DTW classifier without adaptation remains almost constant over time, the accuracy with adaptation improves. The average accuracy in the final bin of size 150 without adaptation was observed to be 62%, while that with adaptation was observed to be 98%. This clearly shows that adaptation allowed the shape models to be morphed to account for the new styles coming from the new samples during testing.

**Experiment 2: Comparison with nearest-neighbor classifier using *AddAndLvq* adaptation strategy** We compared the performance of Active-DTW classifier using the proposed adaptation framework, with a nearest-neighbor classifier using the *AddAndLvq* adaptation strategy (described in Section-3). The *AddAndLvq* strategy was chosen for comparison since it is one of the best performing adaptation strategies in terms of recognition accuracy, described in Vuori's work. Further, it is similar to our strategy in that it adds the sample to the model (i.e. prototype set) if the sample is incorrectly recognized, and morphs the model, if the sample is correctly recognized.

As the adaptation strategy of Active-DTW supports adaptation to begin with zero training samples, no initial training of Active-DTW was performed, while 2 number of training samples were used to train nearest neighbor classifier. For adaptation, 3096 samples (20 samples per character) were presented randomly across classes.

The accuracies were computed over bins of 50 samples, with an overlap of 10 samples and final bin of 100 samples. It can be observed from Figure-2 that the performance of the Active-DTW classifier using the proposed adaptation framework is comparable with that of the NN-classifier using the *AddAndLvq* adaptation strategy.

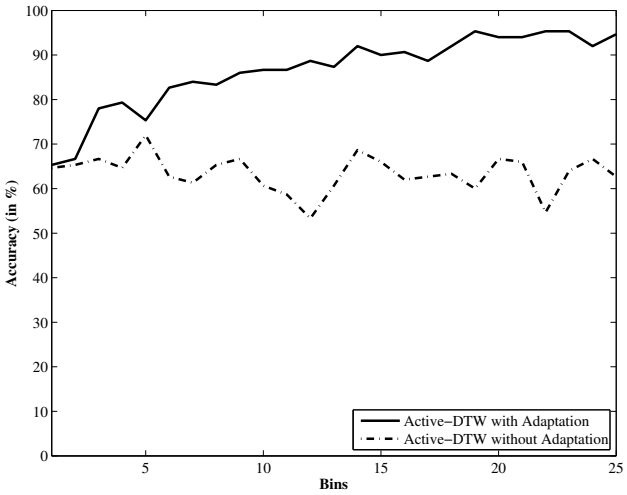


**Figure 1. Recognition performance of Active-DTW with adaptation**

Although the accuracies obtained by the two classifiers are comparable, the memory requirements for model data for Active-DTW are significantly smaller. It can be observed from Figure-3 that both the rate of increase as well as final size of the model data for the NN-classifier are significantly greater that of Active-DTW. This is because *AddAndLvq* adds samples on every mis-recognition, while adaptation of Active-DTW generally adapts the existing Active Shape Models, and only occasionally adds it as a free sample. Time taken to adapt the Active-DTW classifier was observed to be significantly lesser than time taken using *AddAndLvq* strategy for NN-classifier. While the adaptation framework for Active-DTW took 841 seconds for 3096 samples, *AddAndLvq* took 4334 seconds on the same processor.

## 6. Conclusions

A novel framework to adapt the Active-DTW classifier has been proposed in the paper. Such an adaptation framework is particularly useful for rapidly deploying recognizers for new scripts, since the initial requirements for training data can be very small. A considerable improvement in accuracy of the Active-DTW over time was shown empirically using the proposed adaptation strategy. Also, the performance was shown to be comparable with that of the *AddAndLvq* strategy used with a nearest neighbor classifier. The memory requirements for the model data file for Active-DTW recognizer and time taken for adaptation was

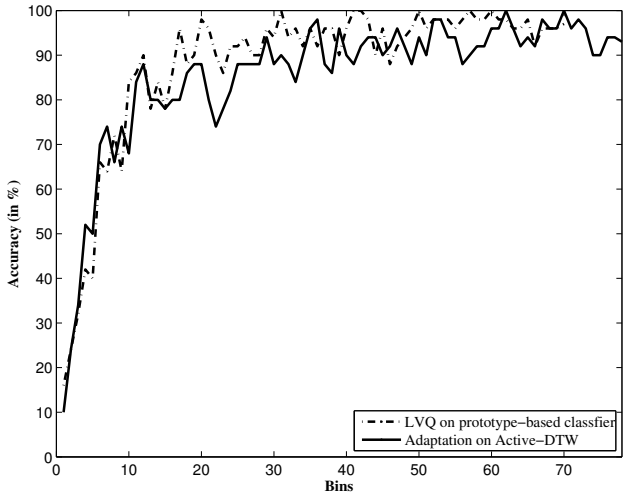


**Figure 2. Comparison of nearest-neighbor and Active-DTW classifiers using respective adaptation schemes**

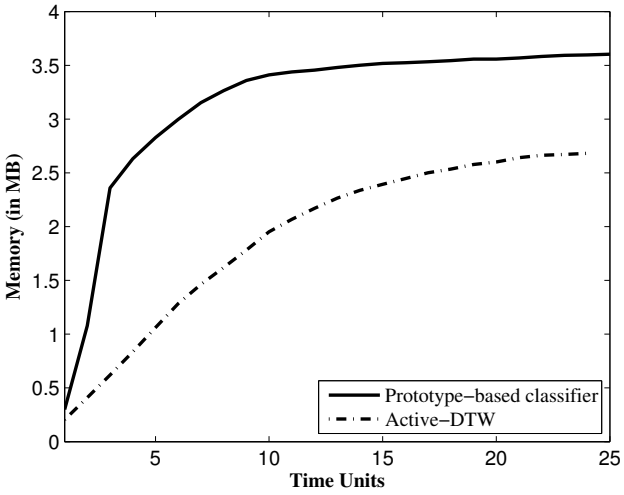


**Figure 3. Comparison of memory requirements of nearest-neighbor and Active-DTW classifiers**

shown to be significantly less as compared to the nearest-neighbor classifier.

As future work, we plan to carry out more experiments and show results on other scripts. We also plan to create better models for the classes to capture discriminative content of different characters.

## References


[1] Lipi Toolkit. `http://lipitk.sourceforge.net/`.

[2] HP Labs Isolated Handwritten Tamil Character Dataset, 2006. `http://www.hpl.hp.com/india/research/penhw-interfaces-1linguistics.html`.

[3] T. Cootes. Active Shape Models - their training and application. *Image Processing and Analysis*, pages 223–248, 2000.

[4] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active Shape Models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, January 1995.

[5] T. F. Cootes, C. J. Taylor, and Lanitis. Statistical models of appearance for medical image analysis and computer vision. *SPIE Medical Imaging*, pages 236–248, 2001.

[6] T. F. Cootes, C. J. Taylor, A. Lanitis, and T. Ahmed. Classifying variable objects using a flexible shape model. *Image Processing and its Applications*, 6(4):70–74, July 1995.

[7] P. M. Hall, D. Marshall, and R. R. Martin. Incremental eigenanalysis for classification. *British Machine Vision Conference*, pages 286–295, 1998.

[8] M. Kass, A. Witkins, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal on Computer Vision*, 1(4):259–268, 1987.

[9] T. Kohonen. Self-organizing maps. *Information Sciences*, 30, 1997.

[10] J. Laaksonen, J. Hurri, E. Oja, and J. Kangas. Comparison of adaptive strategies for on-line character recognition. *International Conference on Artificial Neural Networks*, pages 245–250, 1998.

[11] M. Sridhar, D. Mandalapu, and M. Patel. Active-dtw : A generative classifier that combines elastic matching with active shape modeling for online handwritten character recognition. *International Conference on Frontiers in Handwriting Recognition*, 99(7):1–100, November 1999.

[12] M. Szummer and C. M. Bishop. Discriminative writer adaptation. *International Conference on Document Analysis and Recognition*, 1:31–35, January 2003.

[13] V. Vuori, J. Laaksonen, E. Ojam, and J. Kangas. Controlling on-line adaptation of a prototype-based classifier for handwritten characters. *International Conference on Pattern Recognition*, 2:331–334, 2000.

[14] V. Vuori, J. Laaksonen, E. Ojam, and J. Kangas. Experiments with adaptation strategies for a prototype-based recognition system for isolated handwritten characters. *International Journal on Document Analysis and Recognition*, 3(3):150–159, 2001.

[15] K.-W. Wan, K.-M. Lam, and K.-C. Ng. An accurate Active Shape Model for facial feature extraction. *Pattern Recognition Letters*, 15(26):2409–2423, November 2005.