Best Student Paper Award

Gaussian Process Style Transfer Mapping for Historical Chinese Character Recognition

Jixiong Feng[a] , Liangrui Peng[a] and Franck Lebourgeois[b]

[a]Tsinghua National Laboratory for Information Science and Technology,
Dept. of Electronic Engineering, Tsinghua University, Beijing, China;
[b]LIRIS, Université de Lyon, France

ABSTRACT

Historical Chinese character recognition is very important to larger scale historical document digitalization, but is a very challenging problem due to lack of labeled training samples. This paper proposes a novel non-linear transfer learning method, namely Gaussian Process Style Transfer Mapping (GP-STM). The GP-STM extends traditional linear Style Transfer Mapping (STM) by using Gaussian process and kernel methods. With GP-STM, existing printed Chinese character samples are used to help the recognition of historical Chinese characters. To demonstrate this framework, we compare feature extraction methods, train a modified quadratic discriminant function (MQDF) classifier on printed Chinese character samples, and implement the GP-STM model on Dunhuang historical documents. Various kernels and parameters are explored, and the impact of the number of training samples is evaluated. Experimental results show that accuracy increases by nearly 15 percentage points (from 42.8% to 57.5%) using GP-STM, with an improvement of more than 8 percentage points (from 49.2% to 57.5%) compared to the STM approach.

Keywords: Gaussian process, style transfer mapping, historical Chinese character recognition

1. INTRODUCTION

Historical Chinese character recognition is very important to larger scale historical document digitalization and retrieval.[1–3] Although the accuracy of printed and handwritten Chinese character recognition has improved significantly,[4,5] the recognition of characters from historical documents is still difficult.[6] Reasons include a lack of adequate labeled training samples, large variance in writing styles across documents, and degraded character images due to age and inappropriate preservation.[2,6] There are, however, extensive databases of modern printed and handwritten characters, such as THOCR,[7] HCL2000,[8] HIT-MW[9] and CASIA-HWDB.[10] To make use of these data for historical Chinese character recognition, several methods can be considered. A typical approach is to find common features between characters from historical documents and printed characters, as done in multitask learning.[11–13] Another approach is to view this problem in the transfer learning framework, and project from one domain to another,[14,15] as is widely used in speech recognition.[15]

Transfer learning uses the idea that both historical and modern character domains share the same classes, with a difference in their styles. If a projection can be found to transform the style of one domain to the other domain, then the two domains can share one classifier. Style transfer mapping (STM) has been proposed to solve this problem.[6] STM was first applied in online handwritten Chinese character recognition.[16] By a linear transformation, writer-specific class-independent features were mapped to a style-free space, and then recognized by a writer-independent classifier. STM was a linear regression model; however, for characters from historical documents, linear models may be insufficient to handle severe degradation or noise. In this work we incorporate a more powerful model, the Gaussian process model[17] to extend the linear STM framework.

The Gaussian process model is a probabilistic discriminative model for regression.[18–20] In this approach, the transformation from one domain to the other domain is non-linear. Linear regression models have been shown

Further author information: (Send correspondence to Jixiong Feng)
Jixiong Feng: E-mail: fjx13@mails.tsinghua.edu.cn
Liangrui Peng: E-mail: penglr@tsinghua.edu.cn,
Franck Lebourgeois: E-mail: franck.lebourgeois@insa-lyon.fr

Document Recognition and Retrieval XXII, edited by Eric K. Ringger, Bart Lamiroy
Proc. of SPIE-IS&T Electronic Imaging, Vol. 9402, 94020D © 2015 SPIE-IS&T
CCC code: 0277-786X/15/$18 · doi: 10.1117/12.2076119

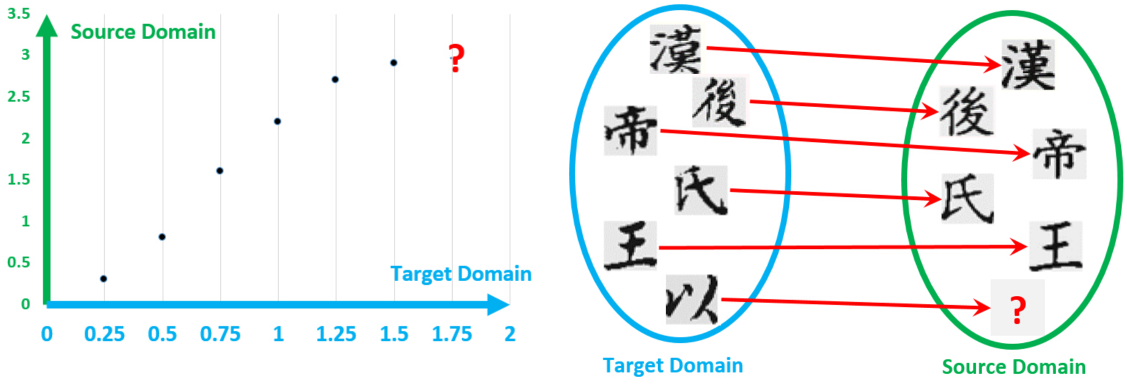
to be special cases of the Gaussian process model.[18] In addition, the parameters of Gaussian process can be learned from multiple tasks,[21,22] and kernels of the Gaussian process can also be learned.[23]

In this paper, we propose a novel Gaussian process based style transfer mapping (GP-STM) model for historical Chinese character recognition. In accordance with customary notation in transfer learning, we denote *the target dataset* as the set of feature vectors of characters from Dunhuang historical documents, while *the source dataset* is composed of feature vectors from printed traditional Chinese characters. A small subset of the target dataset, called *the STM training set*, is used to train the coefficients of the GP-STM. The remaining subset of the target dataset is referred to as *the STM test set*.

The rest of this paper will be organized as follows: Section 2 reviews the related preliminary theoretical concepts, while section 3 presents the framework of our GP-STM model and discusses parameter selection. Section 4 details some experiments based on our model and reports the results, and section 5 gives the conclusions and future work.

2. PRELIMINARY

In this section, we briefly review style transfer mapping (STM) and propose our model, Gaussian Process STM (GP-STM). The key idea of STM and GP-STM is to find a mapping method from a target dataset to a source dataset, which can be viewed as a multivariate regression problem $\mathbf{s} = f(\mathbf{t})$, as shown in Fig. 1. In STM, the transformation f is linear, while in GP-STM it is non-linear. The progression from STM to GP-STM follows naturally from a kernel function perspective. The following subsections will illustrate this evolution from linear to non-linear, going from basis functions to kernel functions, and finally using a Gaussian process to form GP-STM.



(a) (b)

Figure 1. Examples of regression and multivariate regression. **(a)** A typical example of a regression problem. Six (s, t) pairs are given and a new s_* is needed when $t_* = 1.75$. This can be solved by a linear or non-linear regression method. **(b)** An STM problem which can be viewed as a multivariate regression problem. Both inputs and outputs are multi-dimensional. Here inputs are characters from the target domain and outputs are from the source domain. The Chinese characters mean *the dessert, after, the emperor, name, king* and *from* from up to down (they also have other meanings).

2.1 Linear Style Transfer Mapping

Linear STM assumes that different writing styles can be transformed by linear, or affine transformation.[16] The feature vector for character i in the source style (for example written by Steven) is denoted $\mathbf{s}_i$; in the target style (for example written by Tom) it is denoted $\mathbf{t}_i$. Let n be the number of style pairs $(\mathbf{s}_i, \mathbf{t}_i), i = 1, ..., n$. The *Source dataset* is defined as

$$\mathcal{S} = \{\mathbf{s}_i \in \mathbb{R}^d \,|\, i = 1, ..., n\}\,, \tag{1}$$

and the *target dataset*

$$\mathcal{T} = \{\mathbf{t}_i \in \mathbb{R}^d \,|\, i = 1, ..., n\}\,, \tag{2}$$

where d is the length of the feature vector. For any $\mathbf{t}_i \in \mathcal{T}$, we want to identify a simple linear transformation, which can project $\mathbf{t}_i$ to $\mathbf{s}_i \in \mathcal{S}$ with confidence f_i, namely

$$\mathbf{s}_i = \mathbf{A}\mathbf{t}_i + \mathbf{b}\,. \tag{3}$$

Coefficients $\mathbf{A}$ and $\mathbf{b}$ are limited by the modified sum-of squared error function with regularization terms

$$\min_{A \in \mathbb{R}^{d \times d}, b \in \mathbb{R}^d} \sum_{i=1}^{n} f_i \|\mathbf{A}\mathbf{t}_i + \mathbf{b} - \mathbf{s}_i\|^2 + \beta\|\mathbf{A} - \mathbf{I}\|_F^2 + \gamma\|\mathbf{b}\|^2\,, \tag{4}$$

where $\|\cdot\|_F$ is the matrix Frobenius norm, and β and γ are hyperparameters. This optimization problem has a closed-form solution for $\mathbf{A}$ and $\mathbf{b}$

$$\mathbf{A}^* = \frac{\sum\limits_{i=1}^{n} f_i \mathbf{s}_i \mathbf{t}_i^T - \frac{1}{\hat{f}}\hat{\mathbf{s}}\hat{\mathbf{t}}^T + \beta\mathbf{I}}{\sum\limits_{i=1}^{n} f_i \mathbf{t}_i \mathbf{t}_i^T - \frac{1}{\hat{f}}\hat{\mathbf{t}}\hat{\mathbf{t}}^T + \beta\mathbf{I}}, \quad \mathbf{b}^* = \tfrac{1}{\hat{f}}(\hat{\mathbf{s}} - \mathbf{A}^*\hat{\mathbf{t}})\,, \tag{5}$$

where $\mathbf{I}$ is the identity matrix and

$$\hat{f} = \sum_{i=1}^{n} f_i + \gamma, \quad \hat{\mathbf{s}} = \sum_{i=1}^{n} f_i \mathbf{s}_i, \quad \hat{\mathbf{t}} = \sum_{i=1}^{n} f_i \mathbf{t}_i.$$

2.2 Non-linear STM and Kernel Method

We now illustrate how linear STM can be modified to non-linear STM. For simplicity we first rewrite Eq.(3) as a one-dimensional output $s_i \in \mathbb{R}^1$

$$s_i = \mathbf{w}^T \mathbf{t}_i = \sum_{j=1}^{d} w_j t_{ij}\,. \tag{6}$$

The simplest way to create a non-linear transformation is to replace $\mathbf{t}_i$ with $\boldsymbol{\phi}(\mathbf{t}_i)$, where $\boldsymbol{\phi} = (\phi_1, ..., \phi_M)^T$. $\phi_j (j = 1, ..., M)$ are non-linear *basis functions*, mapping $\mathbb{R}^d$ to $\mathbb{R}^1$. The resulting non-linear form of Eq.(6) is

$$s_i = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{t}_i) = \sum_{j=1}^{M} w_j \phi_j(\mathbf{t}_i)\,. \tag{7}$$

Similarly to Eq.(4), the modified sum-of squared error function with regularization terms is given as

$$\min_{\mathbf{w} \in \mathbb{R}^M} \sum_{i=1}^{n} \|\mathbf{w}^T \boldsymbol{\phi}(\mathbf{t}_i) - s_i\|^2 + \lambda\|\mathbf{w}\|^2\,. \tag{8}$$

This error function's derivative with respect to the coefficient $\mathbf{w}$ will be linear, so the optimal coefficient $\mathbf{w}$ to minimize Eq.(8) has the closed form solution

$$\mathbf{w}^* = \boldsymbol{\Phi}^T(\boldsymbol{\Phi}\boldsymbol{\Phi}^T + \lambda\mathbf{I})^{-1}\mathbf{s}\,, \tag{9}$$

where $\mathbf{s} = (s_1, ..., s_n)^T$ and $\boldsymbol{\Phi} = (\boldsymbol{\phi}(\mathbf{t}_1), ..., \boldsymbol{\phi}(\mathbf{t}_n))^T$ is the $n \times M$ *design matrix.* Defining the *kernel* $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$ as an $n \times n$ symmetric matrix with elements

$$K_{ij} = k(\mathbf{t}_i, \mathbf{t}_j) = \boldsymbol{\phi}(\mathbf{t}_i)^T \boldsymbol{\phi}(\mathbf{t}_j)\,. \tag{10}$$

Eq.(9) can be used to rewrite Eq.(7) as

$$s_i = \mathbf{k}(\mathbf{t}_i)^T(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{s}\,, \tag{11}$$

where $\mathbf{k}(\mathbf{t}_i) = \boldsymbol{\Phi}\boldsymbol{\phi}(\mathbf{t}_i) = (k(\mathbf{t}_1, \mathbf{t}_i), ..., k(\mathbf{t}_n, \mathbf{t}_i))^T$. This solves the non-linear regression problem using the kernel method, and prepares the way to extend to a Gaussian process model.

2.3 Gaussian Process

A Gaussian process (GP) extends a multivariate Gaussian distribution to infinite dimensions,[20] which means that any n variables follow an n-variate Gaussian distribution. The covariance matrix $\mathbf{K}$ is defined by a *kernel function*, such as Eq.(10). A widely used kernel function is the single Gaussian kernel.

Kernel 1. The Single Gaussian Kernel is

$$k(\mathbf{t}_i, \mathbf{t}_j) = \theta_0 \exp\left[\frac{-\|\mathbf{t}_i - \mathbf{t}_j\|^2}{2\theta_1^2}\right] + \theta_2 \delta(\mathbf{t}_i, \mathbf{t}_j)\,, \tag{12}$$

where $\theta_i (i = 1, 2, 3)$ are non-negative parameters and $\delta(\mathbf{t_i}, \mathbf{t}_j)$ is the Kronecker delta function. The covariance matrix $\mathbf{K}$ is composed of all $k(\mathbf{t}_i, \mathbf{t}_j)$, for $i = 1, ..., n$ and $j = 1, ..., n$

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{t}_1, \mathbf{t}_1) & \cdots & k(\mathbf{t}_1, \mathbf{t}_n) \\ \vdots & \ddots & \vdots \\ k(\mathbf{t}_n, \mathbf{t}_1) & \cdots & k(\mathbf{t}_n, \mathbf{t}_n) \end{bmatrix}. \tag{13}$$

Still considering $s_i \in \mathbb{R}^1$ (while $\mathbf{t}_i \in \mathbb{R}^d$ with no changes), a Gaussian process model assumes $\mathbf{s} = (s_1, ..., s_n)^T$ follows the n-variate Gaussian distribution with a mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$

$$\mathbf{s} \sim \mathcal{N}(\boldsymbol{\mu},\ \mathbf{K})\,. \tag{14}$$

Now if we have a new vector $\mathbf{t}_* \in \mathcal{T}$ and want to predict the corresponding $s_* \in \mathcal{S}$, it is assumed that

$$\begin{bmatrix} \mathbf{s} \\ s_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \mu_* \end{bmatrix},\ \begin{bmatrix} \mathbf{K} & \mathbf{K}_*^{\mathbf{T}} \\ \mathbf{K}_* & \mathbf{K}_{**} \end{bmatrix}\right), \tag{15}$$

where

$$\mathbf{K}_* = [k(\mathbf{t}_*, \mathbf{t}_1), ..., k(\mathbf{t}_*, \mathbf{t}_n)]\,,\ \mathbf{K}_{**} = k(\mathbf{t}_*, \mathbf{t}_*)\,. \tag{16}$$

The optimal estimate of s_* will maximize the conditional probability $p(s_*|\mathbf{s})$. This probability follows a multivariate Gaussian distribution

$$s_*|\mathbf{s} \sim \mathcal{N}\left(\mu_* + \mathbf{K}_*\mathbf{K}^{-1}(\mathbf{s} - \boldsymbol{\mu}),\ \mathbf{K}_{**} - \mathbf{K}_*\mathbf{K}^{-1}\mathbf{K}_*^{\mathbf{T}}\right). \tag{17}$$

Thus the best estimation for s_* is the mean value of this distribution

$$s_* = \mu_* + \mathbf{K}_*\mathbf{K}^{-1}(\mathbf{s} - \boldsymbol{\mu})\,. \tag{18}$$

Expanding $\mathbf{s}_i$ from 1 to d dimensions, $\mathbf{s}_i \in \mathbb{R}^d$ defined in Eq.(2), $\mathbf{s}_i (i = 1, ..., n)$ can be written as $(s_{i1}, ..., s_{id})$. Assume each dimension $s_{ij} (j = 1, ..., d)$ has the same covariance matrix $\mathbf{K}$, and denote $\boldsymbol{\mu}_* = (\mu_1, ..., \mu_d)$ to be the mean vector of each dimension. Letting the source matrix $\mathbf{S} = (\mathbf{s}_1^T, ..., \mathbf{s}_n^T)^T$ and mean matrix $\boldsymbol{\Omega}$ be matrices with n rows and d columns, then Eq.(18) will becomes

$$\mathbf{s}_* = \boldsymbol{\mu}_* + \mathbf{K}_*\mathbf{K}^{-1}(\mathbf{S} - \boldsymbol{\Omega})\,. \tag{19}$$

2.4 GP-STM

Based on the results in the previous subsection, we now propose our GP-STM model. Considering Eq.(19), we treat the mean vector $\boldsymbol{\mu}_*$ for rough approximation and the $\mathbf{K}_*\mathbf{K}^{-1}(\mathbf{S}-\boldsymbol{\Omega})$ term as a modification. For our model, the source vector $\mathbf{s}_*$ needs to be close to the target vector $\mathbf{t}_*$, while its style should be changed to fit the source style. Therefore, we set $\boldsymbol{\mu}_*$ to be $\mathbf{t}_*$, and the mean matrix $\boldsymbol{\Omega}$ to be the extension of the mean vector of the source vectors $(\bar{\mathbf{s}}, ..., \bar{\mathbf{s}})^T$. Then the GP-STM estimation of source vector $\mathbf{s}_*$ would be

$$\mathbf{s}_* = \mathbf{t}_* + \mathbf{K}_*\mathbf{K}^{-1}(\mathbf{S}-\bar{\mathbf{S}})\,. \tag{20}$$

$\mathbf{K}^{-1}(\mathbf{S}-\bar{\mathbf{S}})$ can be computed in advance, denoted as matrix $\mathbf{A}$. $\mathbf{K}_*$ is a function of $\mathbf{t}_*$, which can be written as $f(\mathbf{t}_*)$. If we write $\mathbf{t}_*$ as $\mathbf{b}(\mathbf{t}_*)$, then Eq.(20) can be write as

$$\mathbf{s}_* = \mathbf{A}f(\mathbf{t}_*) + \mathbf{b}(\mathbf{t}_*). \tag{21}$$

This GP-STM model is similar to the STM of Eq.(3), except that a non-linear function f is added.

There are many options for kernel functions to choose besides Eq.(12). One example is the exponential kernel.

Kernel 2. The Exponential Kernel

$$k(\mathbf{t}_i, \mathbf{t}_j) = \theta_0 \exp(-\theta_1\|\mathbf{t}_i - \mathbf{t}_j\|) + \theta_2\delta(\mathbf{t}_i, \mathbf{t}_j)\,, \tag{22}$$

is used in the *Ornstein-Uhlenbeck Process* to describe Brownian motion. If we want to incorporate a long-term trend, another example is the double Gaussian kernel.

Kernel 3. The Double Gaussian Kernel is

$$k(\mathbf{t}_i, \mathbf{t}_j) = \theta_0 \exp\left[\frac{-\|\mathbf{t}_i - \mathbf{t}_j\|^2}{2\theta_1^2}\right] + \theta_3 \exp\left[\frac{-\|\mathbf{t}_i - \mathbf{t}_j\|^2}{2\theta_4^2}\right] + \theta_2\delta(\mathbf{t}_i, \mathbf{t}_j)\,, \tag{23}$$

where $\theta_3 \approx 6\theta_1$. The only restriction on the kernel function is that $\mathbf{K}$ must be positive semi-definite for any pair of $\mathbf{t}_i$ and $\mathbf{t}_j$.[18] This gives us more freedom to build our model.

3. FRAMEWORK

3.1 Framework Design

The GP-STM framework in this paper can be considered in the context of transductive transfer learning.[14] Given the source dataset $\mathcal{S}$, a corresponding source classifier C_S, the target dataset $\mathcal{T}$, and a corresponding target classifier C_T, transductive transfer learning aims to improve C_T, using the knowledge in $\mathcal{S}$ and C_S and a subset of labeled data from $\mathcal{T}$.

In our framework (see Fig. 2), the source dataset includes Kaiti font from printed Chinese characters. The STM training set includes a subset of labeled handwritten Chinese characters from Dunhuang historical documents, as seen in Fig. 2. Through GP-STM, we successfully use the source dataset and source classifier to help improve the target classifier.

To add GP-STM in recognition system, we divide target dataset into two parts as described previously. About 5% of the characters are used as the *STM training set* to learn the transformation coefficients, while the remaining 95% forms the *STM test set.* Character selection can be done either randomly, or by random selection of specific class categories from the set. In our experiments, we will show the results of both approaches.

All input character images in our experiments are binary images resized to 65×65, to allow concentration on character recognition other than pre-processing. These character images are selected from pages of Dunhuang historical documents, which includes over 11,000 images in over 1,400 classes (see Fig. 3). The source classifier is trained by the source dataset. First, three types of features are extracted, namely 392-dimension Weighted Direction Code Histogram (WDCH) features,[24] 416-dimension Local Binary Pattern (LBP) features[25] and 395-dimension Histogram Oriented Gradient (HOG) features.[26] Fisher Linear Discriminative Analysis (FLDA)[27] is used for dimension reduction, reducing the features to 128-dimension. A Modified Quadratic Discriminant Function (MQDF) classifier[28] is used for classification.

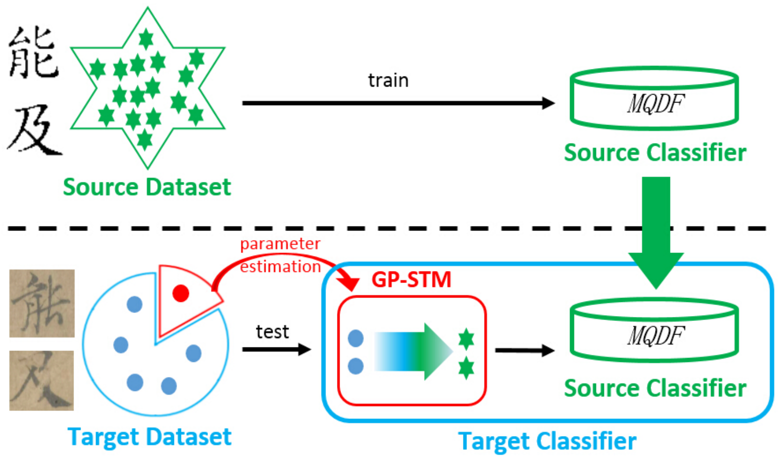


Figure 2. The framework of GP-STM for recognition. Characters in the target dataset are to be recognized (classified). This model is built in several steps. (1) We use the source dataset to train a classifier, named the *source classifier*. (2) The *STM training set* is used to estimate the parameters of the GP-STM, together with the source dataset. (3) The *Target classifier* is composed of the GP-STM model and the source classifier, and is used for recognition. Given a new feature vector from the *STM test set*, it is first transformed by GP-STM, which projects it to source dataset, and then classified using the source classifier. The Chinese characters mean *can* and *until* from up to down (they also have other meanings).

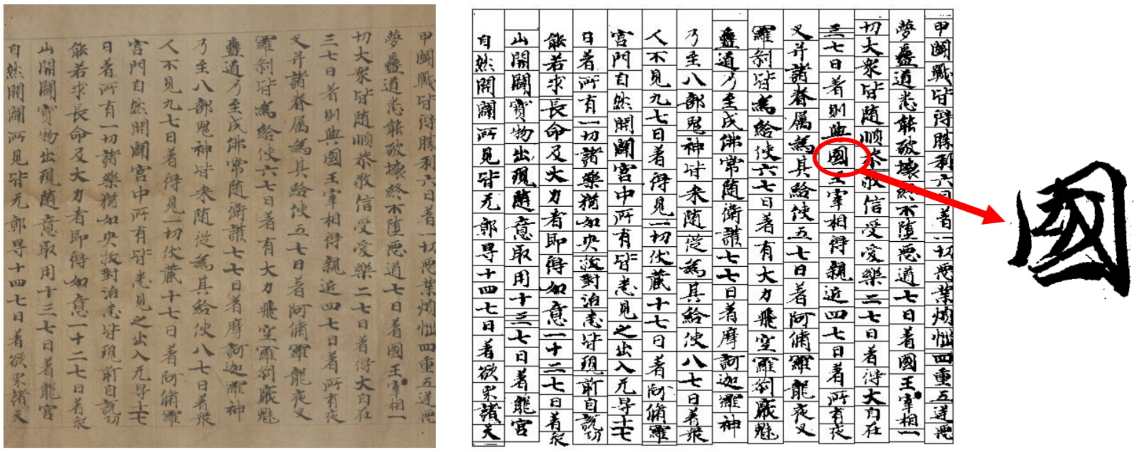


Figure 3. The figure shows how our target dataset comes from. Pages of Dunhuang historical documents are preprocessed (such as denoising and binarization) first, and then character segmentation is used to extract the characters. The characters are resized to 65 × 65 for recognition. Here the Chinese character means *the country*.

3.2 Parameter Selection

To compare with STM, We use $\beta = 0.03$ and $\gamma = 0.01$ in STM method as suggested in Zhang's paper.[16] Because our research is base on supervised learning, confidence $f_i(i = 1, ..., n)$ is set to be $\frac{1}{n}$.

For GP-STM with the kernel described in Eq.(12), the parameter $\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2)^T$. θ_0 is the maximum allowable covariance. θ_1 controls the distance of $\mathbf{s}_i$ and $\mathbf{s}_j$, where larger θ_1 allows more correlation with each other and smaller θ_1 leads to more independence. θ_2 is the noise level, which can also avoid the kernel $\mathbf{K}$ becoming singular. $\boldsymbol{\theta}$ is found by maximizing the posterior $p(\boldsymbol{\theta}|\mathbf{s}, \mathbf{t})$. According to Bayes' theorem, we can alternatively maximizing the log likelihood function

$$\ln p(\mathbf{s}|\mathbf{t},\boldsymbol{\theta}) = -\frac{1}{2}\mathbf{s}^T\mathbf{K}^{-1}\mathbf{s} - \frac{1}{2}\log|\mathbf{K}| - \frac{n}{2}\log 2\pi\,. \tag{24}$$

To accomplish this, we need to find the gradient of the log likelihood function with respect to the parameter vector $\boldsymbol{\theta}$. In general $p(\mathbf{s}|\mathbf{t},\boldsymbol{\theta})$ will not be a convex function, so it can have multiple maxima.[18] Another way is to perform cross-validation. In our model, we empirically suggest $\boldsymbol{\theta}$ to be $(1, 1.5, 0.005)^T$.

For GP-STM with the kernel described in Eq.(23), $\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2, \theta_3, \theta_4)^T$, where two new parameter θ_3 and θ_4 are added. θ_4 in the second item should be larger than θ_1, which means that the kernel takes both short distance and long distance into consideration. θ_1 and θ_3 control the weight between short distance and long distance. Here, we use $\theta_3 = 0.1 \times \theta_1$ and $\theta_4 = 6 \times \theta_2$.

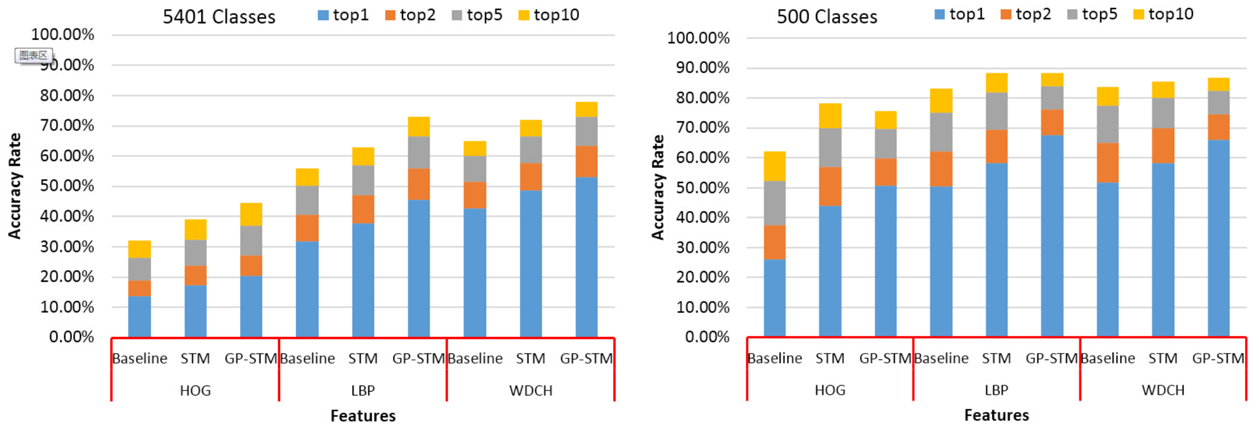
For the exponential kernel in Eq.(22), the meaning of parameters are the same as those of the single Gaussian kernel in Eq.(12). After tuning θ_1 in the range [0,1], we find a proper $\boldsymbol{\theta} = [1, 0.0011, 0]$.

4. EXPERIMENTS

In this section, we conduct experiments on historical Chinese character recognition. First, different types of features are compared on both a smaller dataset and a larger dataset. After that, we compare our GP-STM model to STM and baseline (MQDF classifier directly). Additionally, the influences on recognition accuracy due to changes in the proportion of STM training set, the kernel types, and parameter values, are shown. Finally, we use pixels as features directly, and visualize the results of STM and GP-STM.

4.1 STM and GP-STM

In this experiment, we use three types of features, WDCH, LBP and HOG. We select the characters from Dunhuang historical documents as test samples, with a smaller scale dataset of 500 classes. Results of the feature comparison are shown in Fig. 4a.



(a) (b)

Figure 4. Feature comparison. *Baseline* stands for testing with a MQDF classifier directly. *top1, top2, top5* and *top10* mean the recognition result falls in the first 1, 2, 5 and 10 selections. In (a), a smaller dataset with 500 classes is tested. In (b), a larger dataset with 5401 classes is tested.

After this, we test our model on a larger data set. For traditional Chinese characters, the number of level-1 characters is 5401, according to the BIG5 code standard. All 5401 classes are considered, meaning that the model will handle all possible classes since the test samples don't include all 5401 classes. Results of this feature comparison are shown in Fig. 4b.

Results show that the WDCH feature achieves its best classification accuracy of 51.9% in baseline method; LBP feature achieves its best with both STM and GP-STM, in 58.4% and 67.55%. LBP and WDCH features show

Table 1. Accuracy rate in each model. *top1, top2, top5* and *top10* means the result falls in the first 1, 2, 5 and 10 selections.

Rank	Baseline	STM	**GP-STM**
top1	42.78%	49.18%	**52.77%**
top2	51.65%	58.61%	**62.92%**
top5	60.19%	68.03%	**72.52%**
top10	65.10%	72.92%	**77.57%**

small differences. In the test using all 5401 classes, WDCH feature outperforms other two features significantly. In the following experiments, we use only the WDCH feature.

We can also use this experiment to compare different models. The accuracy of the 5401-class experiment with WDCH features are shown in Table 1.

Here *GP-STM* model uses the kernel defined in Eq.(12). From Table 1 we can see that our GP-STM model has a significant improvement over *Baseline* and *STM*. There is a reduction of error rate by about 10% using GP-STM, which supports the effectiveness of our model.

4.2 Influence of Different Kernels

In this experiment, we look at the impact of the kernels used in GP-STM. Single Gaussian kernel (12), double Gaussian kernel (23) and exponential kernel (22) are used. The results are shown in Fig. 5.

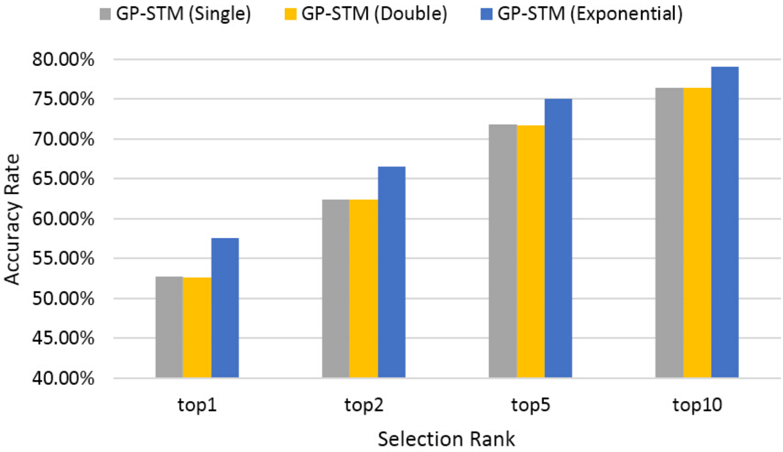


Figure 5. Influence of different kernels on recognition accuracy. *top1, top2, top5* and *top10* mean the recognition results fall in the first 1, 2, 5 and 10 selections respectively.

The exponential kernel outperforms the other two kernels significantly, with a top accuracy of **57.5%**, an improvement of by **14.7** percentage points to *Baseline*, **8.33** percentage points to *STM* and **4.83** percentage points to other kernels. There is little difference between single Gaussian kernel and double Gaussian kernel.

4.3 Influence of the Amount of STM Training Set Data

In previous experiments, we use an STM training set which contained 5% of the total samples for STM/GP-STM training, with the remaining 95% used for STM test set to test our model. Since this is random, the STM training set and STM test set may have a few classes in common.

Here, we make the partition without overlapped classes to see if our model has the ability of generalization. We will tune the proportion of the classes for STM/GP-STM training, and see how this proportion influences the accuracy rates. Experimental results are in Fig. 6.

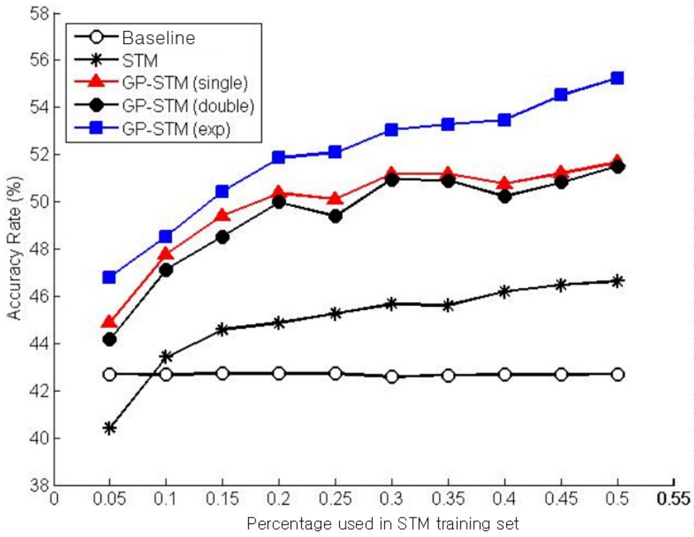


Figure 6. Influence of STM training set percentage.

From Fig. 6 we can see that for STM and GP-STM, the accuracy rates increases with the amount of training data. For *Baseline*, the accuracy rate changes little. Overall, the result shows that knowledge of a few of classes can successfully be transferred to other classes. As the proportion of training data increases, more style details are learned, making the estimated feature vectors more similar to feature vectors from the source dataset.

4.4 Influence of parameter value in GP-STM

We use the exponential kernel in Eq.(22) to check the influence of parameters in GP-STM, as the top performing model. We vary the parameter θ_1 while θ_0 is set to be 1 and θ_2 is 0. The experiment is considered within varying proportions of training data. The results are shown in Fig 7.

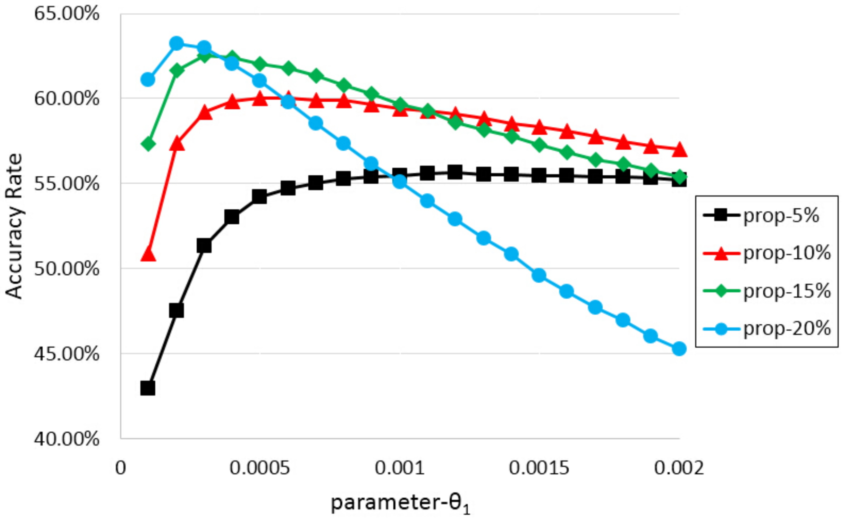


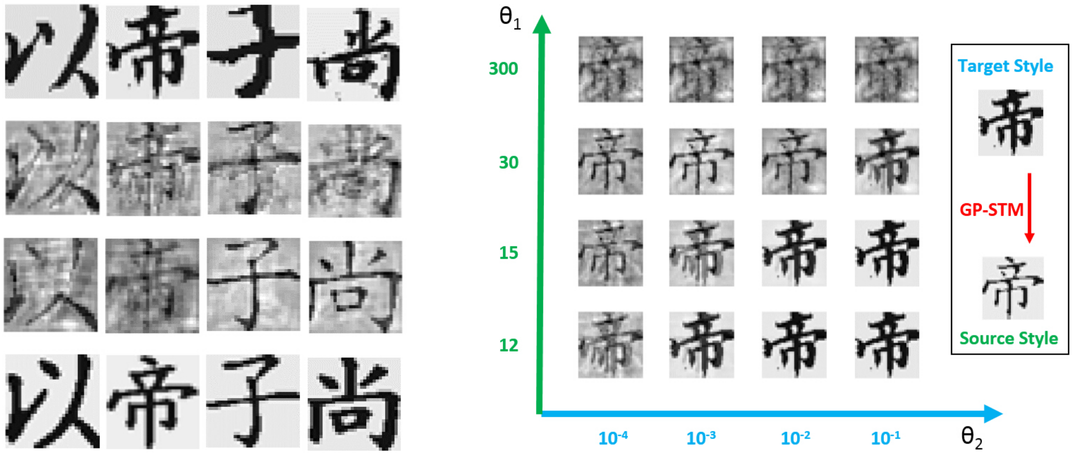
Figure 7. Influence of θ_1 on GP-STM with exponential kernel. *prop-5%* means 5% of samples are used for the STM training set, with *prop-10%, prop-15%, prop-20%* similarly defined.

From Fig 7 we can see that, between the interval $[0, 0.002]$, the accuracy rate has a maximum with respect to θ_1. The maximum value increases together with the amount of training data, while the corresponding optimal

θ_1 moves to zeros. This phenomenon is because a larger proportion of data for training causes the correlation scope to be wider, causing θ_1 to be smaller. Unfortunately, this also indicates that optimum parameter values are data and task dependent and cannot be pre-determined.

4.5 Visualization of STM and GP-STM

In order to visualize how STM and GP-STM transform, we use the pixels of the characters directly. Character images are first resized into 30×30, and then stretched into a 900×1 vectors. According to Eq.(5) and (20), estimations of STM and GP-STM are computed and reshaped back into 30×30. The result of this visualization can be seen in Fig. 8.



(a) (b)

Figure 8. Visualization of STM and GP-STM. **(a)** Comparison of *STM* and *GP-STM*. Characters in the first row are from Dunhuang historical documents, estimated characters using STM are in the second row, estimated characters using GP-STM are in the third row, and characters from printed Chinese dataset are in the fourth row. Note that the left two columns are from the *STM test set*, while the right two columns are from the *STM training set*. The Chinese characters mean *from, the emperor, the son* and *nobility* from left to right (they also have other meanings). **(b)** Influences of θ_1 and θ_2 values of GP-STM for the Chinese character (meaning *the Emperor*).

From Fig. 8a we can see that, in the *STM training set*, GP-STM estimated characters are similar to printed characters. In the *STM test set*, GP-STM estimated characters are blurred more severely, while in detail more similar to printed characters. Overall, GP-STM has better ability for style transfer than STM, and thus has a higher accuracy rate. From Fig. 8b we can see that, just as Sec. 3.2 discussed, larger θ_1 causes more transformation, sometimes with blurring. In contrast, a larger θ_2 keeps the vector more similar to its origin. Proper parameters can transform the vector in target style successfully to source style, such as $\theta_1 = 30$ and $\theta_2 = 10^{-3}$. The process is similar to focusing with a camera, when a proper focus distance would make the scene clearer.

5. CONCLUSION

This paper presents a non-linear STM model based on Gaussian process. Experiments are conducted on Dunhuang historical documents. Compared with the baseline system without STM, GP-STM has a significant improvement by about 15 percentage points compared to baseline, and by 8 percentage points compared with STM. Experiments show that our model has the ability of generalization, and that a few trained classes can benefit the recognition of other classes. We also find that the accuracy rate increases with a larger training set, and illustrate visualization examples for the results of STM and GP-STM. Overall, the proposed GP-STM approaches gives significant improvements in accuracy with a strong ability to generalize.

In our paper, all models are assumed to be supervised. Semi-supervised and unsupervised models will be considered in the future. The optimal parameters and kernels are selected through many experiments, which we hope can be learned automatically in the future. More kernel functions of Gaussian process are to be explored.

Other types of non-linear transformation like Deep Neural Network (DNN) or piecewise linear functions can also be considered.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for suggestions that improved this paper. The first and second authors would like to thank Dr. Michael T. Johnson, Professor from Marquette University, for his constructive suggestions. This research is funded by National Natural Science Foundation of China (No. 61261130590), 973 National Basic Research Program of China (No. 2014CB340506). The second author is also supported by National Natural Science Foundation of China (No. 61032008). The third author is the coordinator of the GuWenShiBie project (Projet Blanc International II ANR-12-IS02-003-03) in France, supported by the ANR (Agence Nationale de la Recherche).

REFERENCES

[1] X. Zhang and G. Nagy, "The CADAL calligraphic database," in *Proc. the 2011 Workshop on Historical Document Imaging and Processing*, pp. 37–42, 2011.

[2] X. Zhang and G. Nagy, "Style comparisons in calligraphy," in *Proc. SPIE 8297, Document Recognition and Retrieval XIX, 82970O*, 2012.

[3] L. Zheng, S. Wang, and Q. Tian, "Lp-norm idf for scalable image retrieval," *IEEE Transactions on Image Processing* **23**(8), pp. 3604–3617, 2014.

[4] F. Yin, Q.-F. Wang, X.-Y. Zhang, and C.-L. Liu, "ICDAR 2013 chinese handwriting recognition competition," in *12th International Conference on Document Analysis and Recognition*, pp. 1464–1470, 2013.

[5] L. Peng, P. Xiu, and X. Ding, "Design and development of an ancient Chinese document recognition system," in *Proc. SPIE 5296, Document Recognition and Retrieval XI*, pp. 166–173, 2003.

[6] B. Li, L. Peng, and J. Ji, "Historical Chinese character recognition method based on style transfer mapping," in *11th IAPR International Workshop on Document Analysis Systems*, pp. 96–100, 2014.

[7] Q. Fu, X. Ding, and C. Liu, "Cascade MQDF classifier for handwritten character recognition," *Journal of Tsinghua University (Science and Technology)* **48**(10), pp. 1065–1068, 2008.

[8] H. Zhang, J. Guo, G. Chen, and C. Li, "HCL2000-A large-scale handwritten Chinese character database for handwritten character recognition," in *10th International Conference on Document Analysis and Recognition*, pp. 286–290, 2009.

[9] T. Su, T. Zhang, and D. Guan, "Corpus-based HIT-MW database for offline recognition of general-purpose Chinese handwritten text," *International Journal of Document Analysis and Recognition* **10**(1), pp. 27–38, 2007.

[10] C.-L. Liu, F. Yin, D.-H. Wang, and Q.-F. Wang, "Online and offline handwritten Chinese character recognition: benchmarking on new databases," *Pattern Recognition* **46**(1), pp. 155–162, 2013.

[11] R. Caruana, *Multitask learning*, Springer, 1998.

[12] B. Su and X. Ding, "Linear sequence discriminant analysis: A model-based dimensionality reduction method for vector sequences," in *IEEE International Conference on Computer Vision*, pp. 889–896, 2013.

[13] G. Tur, "Multitask learning for spoken language understanding," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, **1**, pp. I–I, 2006.

[14] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering* **22**(10), pp. 1345–1359, 2010.

[15] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: transfer learning from unlabeled data," in *Proc. the 24th international conference on Machine learning*, pp. 759–766, 2007.

[16] X.-Y. Zhang and C.-L. Liu, "Writer adaptation with style transfer mapping," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **35**(7), pp. 1773–1787, 2013.

[17] C. Lu and X. Tang, "Surpassing Human-Level Face Verification Performance on LFW with GaussianFace," *arXiv:1404.3840* , 2014.

[18] C. M. Bishop *et al.*, *Pattern recognition and machine learning*, Springer, 2006.

[19] C. E. Rasmussen, *Gaussian processes for machine learning*, MIT Press, 2006.

[20] M. Ebden, “Gaussian processes for regression: A quick introduction,” *Technical report* , 2008. http://www.robots.ox.ac.uk/∼mebden/reports/GPtutorial.pdf.

[21] N. D. Lawrence and J. C. Platt, “Learning to learn with the informative vector machine,” in *Proc. the twenty-first international conference on Machine learning*, pp. 65–73, 2004.

[22] E. Bonilla, K. M. Chai, and C. Williams, “Multi-task Gaussian process prediction,” in *Proc. 20th Ann. Conf. Neural Information Processing Systems*, pp. 153–160, 2008.

[23] A. Schwaighofer, V. Tresp, and K. Yu, “Learning Gaussian process kernels via hierarchical Bayes,” in *Advances in Neural Information Processing Systems*, pp. 1209–1216, 2004.

[24] F. Kimura, T. Wakabayashi, S. Tsuruoka, and Y. Miyake, “Improvement of handwritten Japanese character recognition using weighted direction code histogram,” *Pattern recognition* **30**(8), pp. 1329–1337, 1997.

[25] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(7), pp. 971–987, 2002.

[26] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, **1**, pp. 886–893, 2005.

[27] P. N. Belhumeur, J. P. Hespanha, and D. Kriegman, “Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(7), pp. 711–720, 1997.

[28] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, “Modified quadratic discriminant functions and the application to Chinese character recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **9**(1), pp. 149–153, 1987.