Classifying Trajectory Sequences of Handwritten Letters

COMP9418 Advanced Machine Learning

Mattia Gentil z5178386, Danial Khosravi z5054176, Ryan McKay z5060961

October 24, 2017

1 Abstract

In this report we describe a Hidden Markov Model (HMM) Classifier for handwritten character classification using the sequenced 'Character Trajectories' dataset. Our approach has focused on methodical tuning and validation of a mixture of C (number of target classes) class conditional generative HMM's, with Maximum A Posteriori (MAP) estimation used to make the final class prediction. The initial implementation of our HMM Classifier resulted in a 3-fold validation error rate (ER) of 0.0871. After data cleaning manipulation and tuning of model hyper parameters we achieved a validation ER of 0.0166.

2 Introduction

The task of classification of handwritten characters has been addressed by the research community for a couple of decades by now and perfect character recognition is known to be a challenging task due to the huge diversity of handwriting styles of different people.

Previous work in this field focuses on the classification of images of handwritten characters and is more commonly known as Optical Character Recognition (OCR). Former methods for OCR rely on Hidden Markov Models as the core of their approach [7, 1] to process patches of images directly.

More recent work [8, 3] employs the detection of motion primitives in order to segment a single character in several sections that are common for every handwriting style. Not much work has been carried out for online character recognition using sequential data from the pen movement, even though the temporal order of the strokes or the pen trajectory is shown to provide meaningful information for character recognition [6].

The 'Character Trajectories' data set [4] contains 2858 labeled samples of characters for training with 20 target classes. Each character is represented as a 109-205 long sequence of values in $\mathbb{R}^3$ containing x,y coordinate trajectories and pen tip force differential collected at a 200 Hz sampling rate. Character Trajectories data has been numerically differentiated, Gaussian smoothed and normalized in advance.

HMM's appear among the best approaches adapted to sequences analysis because of both their ability to deal with sequences of variable lengths, and their power to model the dynamic of a phenomenon described by a sequence of events [2].

Before fitting the model, we preprocess the sequences by centering and rescaling to eliminate variability due to translation and size differences [9]. We also prove that a low-pass filtering step is highly beneficial to the classifier's performance, removing noise and standardizing character shapes.

Next, a generative class-conditional Hidden Markov Model is trained on the preprocessed sequences using the Viterbi algorithm and hyperparameter selection is carried out using 3-fold cross validation. Inference is hereby achieved via MAP probability estimation.

3 Model

We present a description of a Hidden Markov Model Classifier that uses a MAP estimation from 20 class conditional discrete time, discrete state Hidden Markov Models (HMM) with Gaussian emissions. We describe how the model functions and how it models the data.

We have the data $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\}$ where each $\boldsymbol{x}^{(n)}$ is a sequence $\boldsymbol{x}^{(n)} = \boldsymbol{x}_1^{(n)}, \boldsymbol{x}_2^{(n)}, \ldots, \boldsymbol{x}_T^{(n)}$, with N training sequences and $T^{(n)}$ length of the sequence n. Each observation $\boldsymbol{x}_t^{(n)}$ is a column vector $\boldsymbol{x}_t^{(n)} = [x_{t,1}^{(n)}, \ldots, x_{t,I}^{(n)}]^T \in \mathbb{R}^I$ of real values, with I being the number of input variables. We will omit the superscript of $\boldsymbol{x}^{(n)}$ when talking about a generic sequence in our dataset. For all $\boldsymbol{x}^{(n)}$ we have target classes $\boldsymbol{y} = \{y^{(1)}, \ldots, y^{(N)}\}$ such that $y^{(n)} \in \{1, \ldots, C\}$, therefore C is the number of target classes .

Our model approximates each class conditional probability $p(\boldsymbol{x}|y = c)$ for $c = 1, \ldots, C$ with a single HMM. The emission probabilities of each HMM given the current state are approximated as a Gaussian distribution, the prior probabilities are assumed to be categorical random variables

Now consider single a discrete time, discrete state HMM with continuous observed values as described above with hidden states $\boldsymbol{z} = \boldsymbol{z}_1, ..., \boldsymbol{z}_T$ such that $z_t \in \{1, ..., Z\}$. By taking advantage of conditional independence of the first order HMM, the corresponding joint distribution is comparable to the observation model described by *Murphy 2012* [5].

$$x_t|z_t \sim \mathcal{N}(\mu_c, \Sigma_c), \quad y_c \sim Cat(\pi_c), \quad z_t \sim Cat(\boldsymbol{\sigma}_s)$$

$$p(\boldsymbol{z}, \boldsymbol{x}) = p(\boldsymbol{z})p(\boldsymbol{x}|\boldsymbol{z}) = p(z_1)p(\boldsymbol{x}_1|z_1) \prod_{t=2}^{T} p(z_t|z_{t-1})p(\boldsymbol{x}_t|z_t) \tag{1}$$

20 HMM's were trained on the class conditional data sets resulting in the generative classifier described above. The classifier predicts the class corresponding to the class conditional HMM with the highest log-likelihood.

Our data was preprocessed before it is used for learning and inference, using only methods that significantly improved the cross validation (CV) error rate. Rescaling and low-pass filtering were applied, providing a substantial performance boost at a low cost, both in terms of computation and model complexity.

Deepu et al. [9] describe a method of rescaling that performs a linear mapping the coordinates to the [0,1] range by transforming the bounding box[1] to a fixed size. This is done for all the sequences $n = 1, \ldots, N$, for all the time steps $t = 1, \ldots, T$ and for all the input vector dimensions $i = 1, \ldots, I$.

$$x_{t,i}^{(n),\text{rescaled}} = \frac{x_{t,i}^{(n)} - \min_{\tau,\eta} x_{\tau,i}^{(\eta)}}{\max_{\tau,\eta} x_{\tau,i}^{(\eta)} - \min_{\tau,\eta} x_{\tau,i}^{(\eta)}}$$

One last step of preprocessing that we found useful to improve the model accuracy is low pass filtering. After several rounds of cross validation we settled on applying a Butterworth 4th order filter to the temporal signals $\boldsymbol{x}_{[1:T],i}^{(n)}$ for all $n = 1, \ldots, N$ and for all $i = 1, \ldots, I$.

[1]The rectangular region that bounds the geometric space occupied by the sequence of input trajectories and therefore the character.

4 Inference

For predicting class y^* for a new observed sequence $\boldsymbol{x}^*$ we calculate a MAP estimate given by Bayes' Rule

$$p(y^*|\boldsymbol{x}^*, \boldsymbol{X}) = \frac{p(\boldsymbol{x}^*|y^*, \boldsymbol{X})p(y^*|\boldsymbol{X})}{p(\boldsymbol{x}^*|\boldsymbol{X})} = \frac{p(\boldsymbol{x}^*|y^*, \boldsymbol{X})p(y^*|\boldsymbol{X})}{\sum_{c\in C} p(\boldsymbol{x}^*|y^* = c, \boldsymbol{X})p(y^* = c|\boldsymbol{X})}$$

We predict the class for a new data point by computing

$$\begin{aligned}\hat{y}^* &= \underset{c\in C}{\operatorname{argmax}} \left(\log\left[p(y^* = c|\boldsymbol{x}^*, \boldsymbol{X})\right]\right)\\ &= \underset{c\in C}{\operatorname{argmax}} \left(\log\left[p(\boldsymbol{x}^*|y^* = c, \boldsymbol{X})\right] + \log\left[p(y^* = c|\boldsymbol{X})\right] - \log\left[\sum_{c'\in C} p(\boldsymbol{x}^*|y^* = c', \boldsymbol{X})p(y^* = c'|\boldsymbol{X})\right]\right)\end{aligned}$$

The term $\log[p(\boldsymbol{x}^*|y^* = c, \boldsymbol{X})]$ is computed by the c-th class conditional HMM by summing over all hidden paths $z_{1:T}$. This is also known as the probability of the evidence [5].

$$p(\boldsymbol{x}^*_{1:T}) = \sum_{\boldsymbol{z}_{1:T}} p(\boldsymbol{z}_{1:T}, \boldsymbol{x}^*_{1:T}) = \sum_{\boldsymbol{z}_{1:T}} p(z_T, \boldsymbol{z}_{1:T-1}, \boldsymbol{x}^*_T, \boldsymbol{x}^*_{1:T-1}) = p(\boldsymbol{x}^*_T|z_T) \sum_{z_{T-1}} p(z_T|z_{T-1})p(z_{T-1}, \boldsymbol{x}^*_{T-1})$$

Firstly, the per-component log probability under the model $p(z_t|x_t)$ is computed. Then a forward pass is performed in order to get the class conditional data log likelihood $\log[p(\boldsymbol{x}^*|y^* = c, \boldsymbol{X})]$. Instead, the class prior probabilities $p(y^* = c|\boldsymbol{X})$ are obtained by data counting

$$p(y^* = c|\boldsymbol{X}) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}(y^{(n)} = c)$$

5 Parameter Estimation

The joint distribution described by 1 is parameterized by

$$A_{i,j} = p(z_t = i|z_{t-1} = j), \quad \pi_i = p(z_1 = i), \quad b_j = p(x_t|z_t = j) = \mathcal{N}(\boldsymbol{x}; \mu_j, \Sigma_j)$$

Where Σ_j is a positive definite covariance matrix, which we restrict to be a diagonal matrix as a tradeoff between complexity of the model and training speed.

This parameterization defines $\boldsymbol{\pi}$ as the initial state or prior distribution of discrete hidden variables z_t, $\boldsymbol{A}$ is the transition matrix between hidden variables z_t and B are the parameters of the data class conditional densities [5]. The initial distribution $\boldsymbol{\pi}$ and the state transition matrix $\boldsymbol{A}$ are initialized statically and then updated when training is performed. In particular, we constrain our model to start from the first state by setting

$$\boldsymbol{\pi} = [1, 0, \ldots, 0]$$

For each $z_t = s$ we only allow z_{t+1} to be either s or $s + 1$, hence introducing a linear progression for the hidden states. This is done by initializing

$$\boldsymbol{A} = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & \ldots & 0 & 0\\ 0 & 0.5 & 0.5 & 0 & \ldots & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & & \vdots & \vdots\\ 0 & 0 & 0 & \ldots & 0 & 0.5 & 0.5\\ 0 & 0 & 0 & \ldots & 0 & 0 & 1 \end{bmatrix}$$

Each HMM is then trained on the class conditional data and the Viterbi algorithm is employed in order to optimize the fit of $\boldsymbol{\pi}$ and $\boldsymbol{A}$ to the training data.

Given z_t are not observed, the EM algorithm is employed to approximate the parameters of the Gaussian emission probabilities [5], that are $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$, for $j = 1, \ldots, S$ where S is the number of possible hidden states.

The hyperparameters S and the cutoff frequency f of the low-pass filter were settled by running several rounds of cross-validation and restricting the grid search at each run. We used a stratified k-fold cross validation framework with $k = 3$ folds and maximized the accuracy on validation data in order to pick the best performing parameters, which turned out to be $S = 10$, $f = 3$ Hz.

6 Results

Our experimental methodology focused on achieving a high 3-fold stratified cross-validation accuracy while maintaining a low model complexity. We addressed sources of misclassification by analyzing the confusion matrix of the predictions. Table 1 contains the summary of results of our experiments including our final model.

Our initial baseline model was a HMM classifier described in section 3 with $S = 10$ with 91% accuracy. After cross-validating the S for a wide range of values, $S = 28$ performed well on the non-filtered data achieving an accuracy of just below 93%.

This validation produced a confusion matrix of the models predictions (**Appendix A**) that revealed a primary source of misclassification was confusion between the characters $\{h, m, n, u, w\}$.

The approach for our third model was to set S=28 as the base number of states, for all class conditional HMM's. We used cross validating error rate to handpick the optimized S for the confused characters. This tuning method resulted in an accuracy of over 94%.

Finally, introducing low pass filtering as a further preprocessing step allowed us to reduce the noise in the dataset and build models with lower complexities that yielded higher cross-validation accuracies. After choosing the optimized hyperparameters, as explained in section 5, our final model with $S = 10$ for all the characters achieved an accuracy of over 98% on the filtered data while minimizing model complexity and computation time. Additionally, it should be mentioned that we tried Gaussian Mixture Models (GMM) to classify the data. Although GMMs are not necessary known for modeling sequences, our GMM classifier produced very significant results that are included in **Appendix B** for comparison purposes.

Table 1: Result of experiments: the Error Rate and Mean Negative Log Probability were computed by averaging the results on validation data with 3-fold cross validation. The training time is an average running time of the training process for the model with the entire training set of 1,429 sequences. The evaluation time refers to the average time needed to evaluate a single sequence.

Model	**ER**	**MNLP**	**Training Time (s)**	**Evaluation Time (ms)**
Basic 10 states	0.0871	3.060	35.27	10.56
Basic 28 states	0.0705	1.999	192.05	57.35
Optimized states per character	0.0581	1.834	192.75	54.93
Low pass filtered with 10 states	0.0166	0.695	36.01	11.33

References

[1] Oscar E Agazzi and Shyh shiaw Kuo. Hidden markov model based optical character recognition in the presence of deterministic transformations. *Pattern Recognition*, 26(12):1813 – 1826, 1993.

[2] Rakia Jaziri, Mustapha Lebbah, Younès Bennani, and Jean-Hugues Chenot. *SOS-HMM: Self-Organizing Structure of Hidden Markov Model*, pages 87–94. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[3] Lucas Kovar and Michael Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Trans. Graph.*, 23(3):559–568, August 2004.

[4] M. Lichman. UCI machine learning repository, 2013.

[5] Kevin P Murphy. *Machine learning: a probabilistic perspective.* Cambridge, MA, 2012.

[6] Vu Nguyen and Michael Blumenstein. Techniques for static handwriting trajectory recovery: A survey. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, DAS '10, pages 463–470, New York, NY, USA, 2010. ACM.

[7] HEE-SEON PARK and SEONG-WHAN LEE. A truly 2-d hidden markov model for off-line handwritten character recognition. *Pattern Recognition*, 31(12):1849 – 1864, 1998.

[8] Goutam Sanyal. Handwritten character recognition using hidden markov model segmenting into equivalent basic geometric primitives. *Journal of Science & Management (LJSM)*, 2015.

[9] Deepu Vijayasenan, S Madhvanath, and A.G. Ramakrishnan. Principal component analysis for online handwritten character recognition, 08 2004.

Appendix

A Confusion matrix of states

Table 2: Confusion Matrix for validation data with each class conditional HMM having $S = 28$ hidden states. Precision and recall are reported in the last row and column of the table respectively

	a	**b**	**c**	**d**	**e**	**g**	**h**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**u**	**v**	**w**	**y**	**z**	Recall
a	28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
b	0	26	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0.93
c	0	0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
d	0	0	0	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
e	0	0	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
g	0	0	0	0	0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
h	0	0	0	0	0	0	10	0	0	6	0	0	0	0	0	0	0	3	0	0	0.53
l	0	0	0	0	0	0	0	26	0	0	0	0	0	0	0	0	0	1	0	0	0.96
m	0	0	0	0	0	0	0	0	16	5	0	0	0	0	0	0	0	2	0	0	0.70
n	0	0	0	0	0	0	0	0	4	15	0	0	0	0	0	0	0	2	0	0	0.71
o	0	0	0	0	0	0	0	0	0	0	22	0	0	0	0	0	0	0	0	0	1
p	0	0	0	0	0	0	0	0	0	0	0	24	0	0	0	0	0	0	0	0	1
q	0	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	1
r	0	0	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0	1
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22	0	0	0	0	0	1
u	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	6	0	0	0.73
v	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30	0	0	0	1
w	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	17	0	0	0.85
y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23	0	1
z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	31	1
Prec.	1	1	1	1	1	1	1	1	0.76	0.56	1	0.96	1	1	1	0.94	1	0.53	1	1	

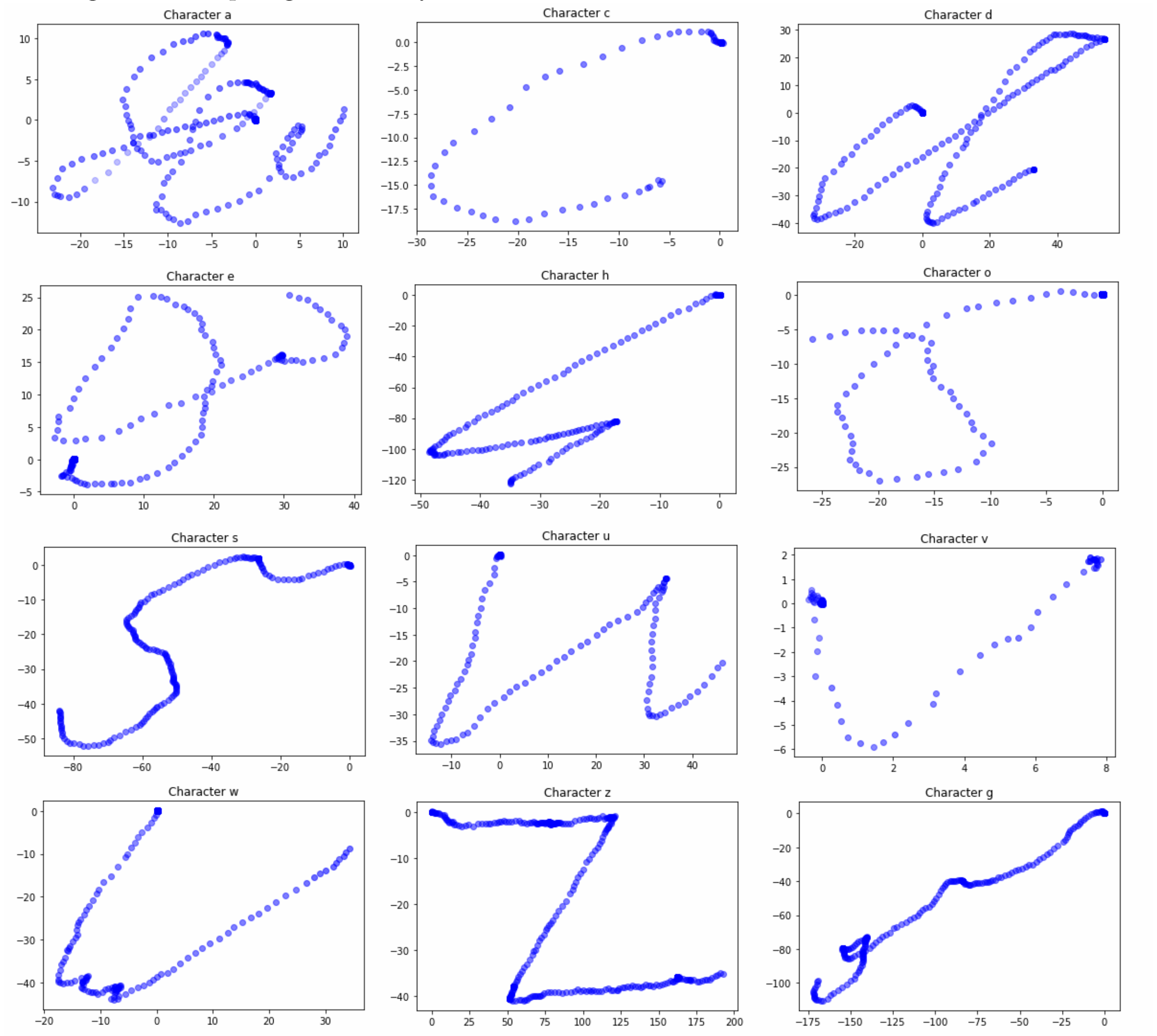
B Gaussian Mixture Model Generative classifier

We created a GMM Generative Classifier (similar to the one used for Assignment 1). Since the sequences have different lengths, we padded the sequences with zeros to achieve a fix length of 250 and stacked the dimension, obtaining an array of shape $(1429, 750)$. This was used as the input to our GMM Generative Classifier. Training the classifier with a diagonal covariance resulted in an average 3-fold cross validation accuracy of 97.48%. A surprising result of 99.50% was obtained when training the classifier with a full covariance.

C Generated digits from HMM sampling

We found some interesting insight on the learning process of HMMs by analyzing which motions were learned and recognized by the class conditional HMMs. Therefore we randomly sampled some signals for trajectories and we plotted them. Some interesting results are reported in Figure 1.

Figure 1: Samples generated by the trained class conditional HMMs with $S = 28$ states.


Character a
Character c
Character d
Character e
Character h
Character o
Character s
Character u
Character v
Character w
Character z
Character g