Signatures in online handwriting recognition

Jeremy Reizenstein, Centre for Complexity Science[*]
with
Ben Graham, Department of Statistics and Centre for Complexity Science
University of Warwick

September 2014

Abstract

We investigate the use of iterated integral signatures as a representation of handwritten characters for improved machine recognition, using data from Pendigits and Assamese. We compare methods for including information about pen-liftings in multi-stroke characters and show that the ink dimension is most useful. We show that much useful information is lost when moving to a rotationally-invariant representation.

1 Introduction

This project aims to identify a methodology which improves the accuracy and efficiency of machine recognition of handwritten characters using the *iterated integral signature*. Online handwriting recognition is the task of identifying the intended characters from the movement of a pen or stylus. This is relevant when characters are written by a user on a touchpad attached to a computer. In this project we are concerned with the classification of single characters, assuming that the input has already been segmented into characters.

Rather than pick a single alphabet and group of writers to study and adapt a method to, we wish to define only a more general algorithm which is capable of being adapted to any alphabet using many samples of that alphabet's characters formed by different writers. This process of adaptation is called **training** or learning. A better algorithm would be one that predicts the characters of additional samples with better accuracy after the same amount of training.

With a touchpad, information about the direction of the pen's movement and the order in which the strokes were made is available. This is in contrast to *optical* or *offline* character recognition where only an image of the formed character is available. The touchpad does not provide information about how much pressure was applied nor the speed of the pen.

The task of recognising characters is an example of a supervised machine learning problem: one where a computer is provided with labelled data and asked to extend the labelling to new data. There are established general-purpose methods of solving such a problem from a well-defined domain, for example $\mathbb{R}^k$ for a fixed k; these are called classification algorithms. The problem of online handwriting recognition discussed here can be approached with one of these established methods if we define a deterministic, easily computed function, a **representation** from input samples as provided by the touchpad. The whole system's process of labelling is in two steps as illustrated below.

$$\text{input} \xrightarrow{\text{representation}} \mathbb{R}^k \xrightarrow{\text{classification}} \text{alphabet}$$

[*]Supported by the Engineering and Physical Sciences Research Council

A good representation should reflect those *features* of the input which often differ from character to character. Such features constitute **useful** information for the classifier. A good representation should include only those aspects of the input which are relevant to identifying a character. Having such a representation should make the training process work well and mean that the identification of characters has high accuracy.

This project aims to find such good choices of the representation, continuing the work of J. Diehl[5] and B. Graham[6] which uses the *iterated integral signature* or *signature* in this task. The signature, defined in Section 3, is a concept which comes from the theory of differential equations driven by rough paths (see e.g. [8]).

This paper is organised as follows: Section 2 introduces what the data looks like as it comes from a touchpad and the two datasets which we are working with. Section 3 describes the signature concept. Section 4 addresses how to transform a character with multiple strokes into a form from which a signature can be taken. Section 5 describes approaches to making a representation rotationally invariant. Section 6 describes the method of testing representations. Section 7 presents the results of our experiments and Section 8 concludes. Appendix A contains some technical remarks about *linear rotational invariants*.

2 Data

In this paper we have used the datasets Pendigits and Assamese, which contain samples of the touchpad output from many different writers of the characters of the alphabet in question. We test our method by splitting the dataset into a set of training data and a set of testing data. We train the algorithm with only the training data, and then use the algorithm to predict the characters for all the testing data. We measure the proportion of these predictions which were correct.

Pendigits [1] consists of handwritten samples of the digits 0–9. It has a customary split between training data and testing data: 7494 characters from 30 different writers form the training data and 3498 samples from 14 different writers form the testing data.

The **Assamese** dataset [3] contains 45 samples, all from different writers, of each of 183 characters. We split this data so that the first 36 of each character are used for training and the last 9 of each character are used for testing.

The data for each character is a list of strokes, where each stroke is given as a series of points. We assume that the pen moved straight between the given points, so that the points are the turning points of a piecewise linear path traced out, and that the pen was lifted between each stroke. The character is scaled, maintaining aspect ratio, so that the centre of its bounding box is the origin and the maximum of its height and width is 2. Thus all the points in the character fit inside a square with x and y coordinates between -1 and 1. As an example, Figure 1 shows a plot of the two strokes of a number '7' from the Pendigits dataset.

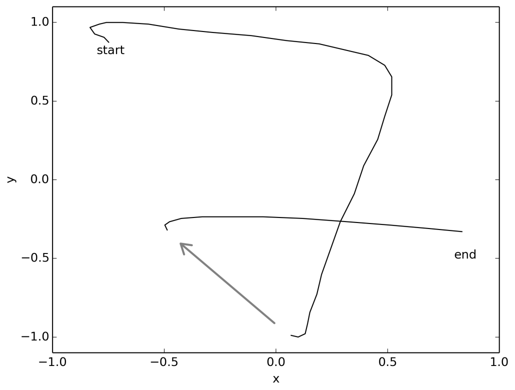

Figure 1: An example from the Pendigits dataset, a number ‘7’

3 Signatures

An iterated integral signature is a sequence of numbers derived from a continuous path. An example of a continuous path in two dimensions is the path traced out by the nib of a pen during a single contact with the paper. The signature is invariant on translations of the path. It is also invariant on reparametrizing the path. It may therefore be useful in defining a good representation function. If two paths are different then their signatures will be different, unless one contains a section where the path exactly retraces itself.[7] A signature cannot distinguish between two paths where the only difference is that one path contains an extra section which is backtracked over. It is difficult for a writer to backtrack precisely, and so every character should be distinguishable through its signature. It is possible that approximate backtracking might lead to numerically unstable signatures.

Parsimonious definition

A path in $\mathbb{R}^d$ (with coordinate axes $x_1, x_2, \ldots, x_d$) can be described by a continuous map $\gamma : [a, b] \to \mathbb{R}^d$ with $\gamma(t) = (\gamma_1(t), \gamma_2(t), \ldots, \gamma_d(t))$. Its **signature** is a function from words written in the alphabet $\{1, 2, \ldots, d\}$ to $\mathbb{R}$, denoted by $X^\gamma_{a,b}$. Some example words are ‘1’, ‘21’ and ‘121’. The signature is defined inductively on the length of the word. The signature of the empty word is defined as 1. If w is a word and $i \in \{1, 2, \ldots, d\}$ then $X^\gamma_{a,b}(wi)$ is defined as $\int_a^t X^\gamma_{a,t}\gamma_i'(t)\,dt$. The restriction of the signature to words of length m is called the mth **level** of the signature. It contains d^m values.

If $a < b < c$ then the result (from [4]) known as **Chen’s identity** states that

$$X^\gamma_{a,c}(i_1 i_2 \ldots i_n) = \sum_{j=0}^{n} X^\gamma_{a,b}(i_1 i_2 \ldots i_{j-1}) X^\gamma_{b,c}(i_j i_{j+1} \ldots i_n). \tag{1}$$

Also, if γ is a straight line then

$$X^\gamma_{a,b}(i_1 i_2 \ldots i_n) = \frac{1}{n!} \prod_{j=1}^{n} (\gamma_{i_j}(b) - \gamma_{i_j}(a)). \tag{2}$$

These two facts between them make it easy to calculate elements of the signature of a piecewise linear path up to any given level.

Alternative explanation

The signature is made up of a series of levels, one for each nonnegative integer. Level m takes values in $(\mathbb{R}^d)^{\otimes m}$, which is a d^m-dimensional real vector space, and consists of the d^m values of integrals of the form

$$\int_a^b \int_a^{t_1} \dots \int_a^{t_{m-2}} \int_a^{t_{m-1}} dx_{i_1}(t_m)\, dx_{i_2}(t_{m-1})\, \dots\, dx_{i_{m-1}}(t_2)\, dx_{i_m}(t_1), \tag{3}$$

where each i_j is allowed to range over values in $\{1, 2, \dots, d\}$.

In this form, the signature is seen to be an element of the tensor algebra $T(\mathbb{R}^d) = \bigoplus_{m=0}^{\infty} (\mathbb{R}^d)^{\otimes m}$. Chen's identity is a convolution and the signature of a line segment is an exponential series.

3.1 Log signature

Let S be the set which consists of level m of the signature of every path in $\mathbb{R}^d$. S is not the whole of the vector space $(\mathbb{R}^d)^{\otimes m}$, although it does *span* $(\mathbb{R}^d)^{\otimes m}$ (see Lemma 8 in [5]). In fact, they form a lower dimensional manifold. The logarithm operation ([10], chapter 3) $\log : T(\mathbb{R}^d) \to T(\mathbb{R}^d)$ is defined by $\log(1 + T) = \sum_{n \geq 1} \frac{(-1)^{n-1} T^n}{n}$. The logarithms of the signatures of all possible paths does form a linear subspace of $T(\mathbb{R}^d)$, and a basis, the Hall basis, can be constructed for it up to level m. The representation of the logarithm of the signature in the Hall basis is called the **log signature**. The log signature of a path up to level m is determined by an easy calculation from its signature up to level m, and the passage to the log signature can be considered an information compression or a dimensionality reduction. We performed the calculation with the aid of the CoRoPa software library [9].

4 Paths from multi-stroke characters

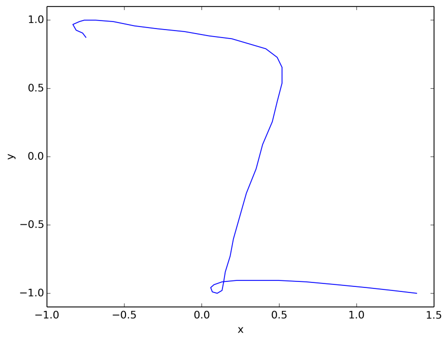

Figure 2: The same example from the Pendigits dataset, with strokes moved so that they join up. This resembles a number '2'.

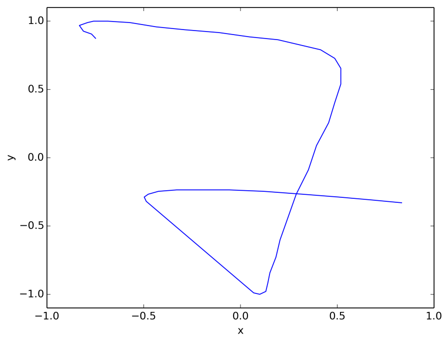

Figure 3: The same example from the Pendigits dataset, with the pen kept on the paper. This still resembles a number '7'.

We aim to generate representations by taking elements from the signatures of paths. Because the number of strokes in a single character can be very variable, it would be an inefficient representation to take some elements of the signature of each stroke and then concatenate them. We want, rather, to form a single path from all the strokes and take elements of its signature. We want to make it easy for the algorithm to learn that the same character might occasionally have sections where the pen might be up or down. Ideally the path would be similar for similarly-intended characters and different otherwise.

The simplest conceivable way to join the strokes into a path would be to form a single curve in two dimensions by starting each successive stroke at the end of the previous stroke. Figure 2 shows what this looks like for the previous number '7'. It resembles a number '2'. In general, this method loses the information about which stroke was where, which we think is usually important. Therefore we reject this method. The simplest method which we consider is to keep the strokes in their original positions and join them with straight line segments. We call this the concatenated character. It is illustrated in Figure 3.

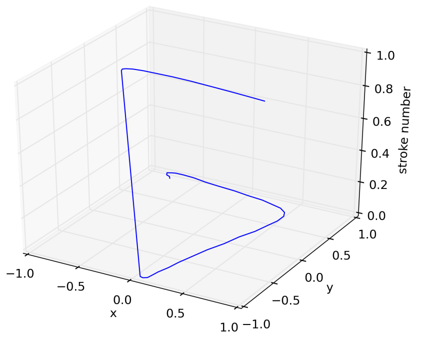

Figure 4: A three dimensional view of the same example from the Pendigits dataset, with a stroke dimension

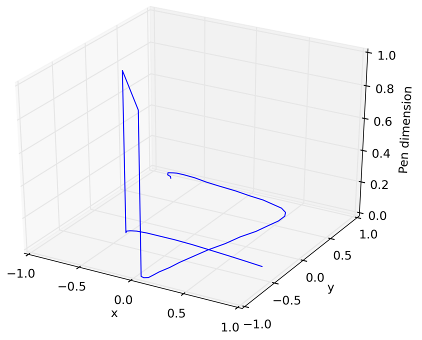

Figure 5: A three dimensional view of the same example from the Pendigits dataset, with a pen dimension

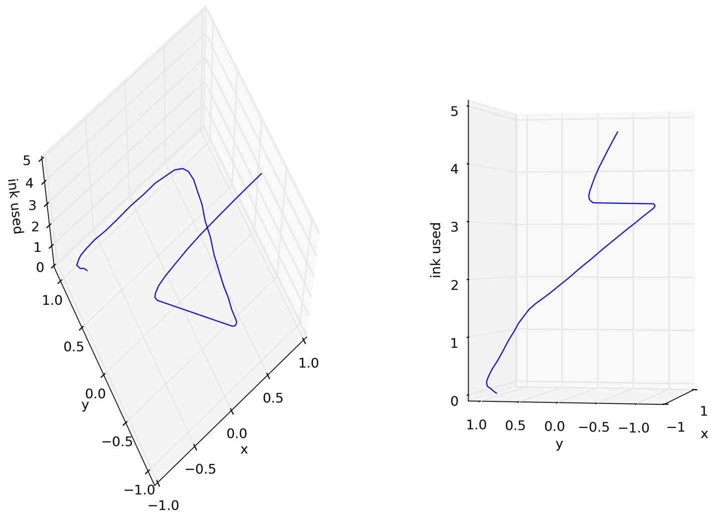

Figure 6: Two different viewpoints of a three dimensional view of the same example from the Pendigits dataset, with an ink dimension.

To preserve the information about when the pen was on the paper, we need a third dimension. One way is to number the strokes and to give every point in the concatenated character the stroke which it is in. This we call the **stroke dimension** and it is illustrated in Figure 4. A second way is the **ink dimension**, where to each point is appended the amount of ink the pen would have released by that point in the character, so that jumps between strokes are identifiable as being in a different direction from equivalent straight strokes. This is difficult to visualise in a two dimensional graph, an attempt is given in Figure 6. We have a choice whether to normalise the ink dimension so that every character uses the same amount of ink or not. A third method is what we call the **pen dimension**, in which we consider the pen position as being up (value 1) or down on the paper (value 0), and moves are either of the pen in the pen dimension or of the pen's location on the plane. This is illustrated in Figure 5.

5 Rotational Invariance

Because people in the real world usually do not need to know which way up a character was written to recognise it, we think that a machine also should not need this information. For example, although numbers '6' and '9' appear only to differ by a rotation, people tend to write them in very different ways and they are recognisable. Irrelevant information in general should not be included in a representation as it could make the machine learning task harder and less robust. Information in the path which tells us which orientation the character was in might fall into that category. Also, some of the variation in handwriting between people, which we want our algorithm to be immune to, might be orientation related. One way to ensure our representation function is rotationally invariant is, as the first step in producing the representation, to rotate the input so that the last point of the character lies exactly to the right of the first point. The effect of this is that the algorithm gets the same input whichever orientation the letter was made. Equivalently, the algorithm does not know which way is 'up' on the characters it sees. When we do this, we use the whole of the first few levels of the signature as the representation, except that we omit the values $X^{\gamma}_{a,b}(2)$, $X^{\gamma}_{a,b}(22)$, $X^{\gamma}_{a,b}(222)$ etc. because they will always be 0 for a path with no total vertical displacement.

In [5] is derived an expression for all the linear combinations of elements of the first m levels of the signature of a two dimensional path which are, for all paths, invariant under rotations of the path. We call these **linear rotational invariants**. Two which have simple interpretations are that $\frac{1}{2}X^{\gamma}_{a,b}(21) - \frac{1}{2}X^{\gamma}_{a,b}(12)$ is the signed area between the curve γ and the straight line from $\gamma(a)$ to $\gamma(b)$ and that $2X^{\gamma}_{a,b}(11) + 2X^{\gamma}_{a,b}(22)$ is the square of the distance from $\gamma(a)$ to $\gamma(b)$[1]. A second way to ensure our representation function is rotationally invariant is to use linear rotational invariants as the signature. This is already proposed and explored in [5].

6 Testing

In testing representations, we train the algorithm with only the training data, and then use the algorithm to predict the characters for all the testing data. The proportion of these predictions which are incorrect is known as the test error.

We needed to choose a classification algorithm to do these tests. The classification algorithm is treated as a black box. We used a fully connected feedforward artificial neural network with three hidden layers each of 300 cells, trained by backpropagation. The training was run for 1001 epochs each of 100 batches of 100 data samples. As a form of data augmentation, we applied elastic distortion to each test sample before using it (so that the network is less likely to become adapted to some features of the data which are destroyed by small rotations and stretches to the data). Training proceeds by taking batches of training samples and optimising the network for it using a small number of steps with a multidimensional optimisation algorithm. We also used 50% dropout in all layers but the first.

7 Results

7.1 Two spatial dimensions

In this section, we consider the usefulness of successive levels of the signature, and the success reductions for rotational invariance. We used the Pendigits dataset alone, where every character has three or fewer strokes, and so the multi-stroke considerations are less important than with the Assamese dataset. The lengths of the representations used and the test error percentages found using each are given in Table 1.

Figure 7 shows that using more levels of the signature reduces the test error, as expected. Using more levels of the signature includes some more information about the curve, and also includes some nonlinear repeats of information in lower levels. The log signature does not contain redundant information in any level, and the (smaller) improvement we see as more of it is used as a representation is exactly the information gleaned from further integrals. For three or more levels, the log signature offers poorer performance than the same number of levels of signature, because of the nonlinear repeated information.

[1]Up to a constant, this is given in [5], page 3

Levels included	1	2	3	4	5	6	7	8	9
Length of signature	2	6	14	30	62	126	254	510	1022
Length of log signature	2	3	5	8	14	23	41		
Length of rotated signature	1	4	11	26	57	120	247	502	
L. of linear rotational invariants		2		5		15		46	
Test error of signature	44.65	19.81	10.75	5.86	4.83	3.09	3.0	2.52	2.29
Test error of log signature	44.74	19.84	12.12	8.81	7.2	6.0	5.92		
Test error of rotated signature	69.01	40.74	13.81	7.12	5.37	4.2	3.97	3.49	
T. e. of linear rotational invariants		40.85		11.01		6.49		6.03	

Table 1: Representation length and minimum test error percentage for concatenated Pendigits

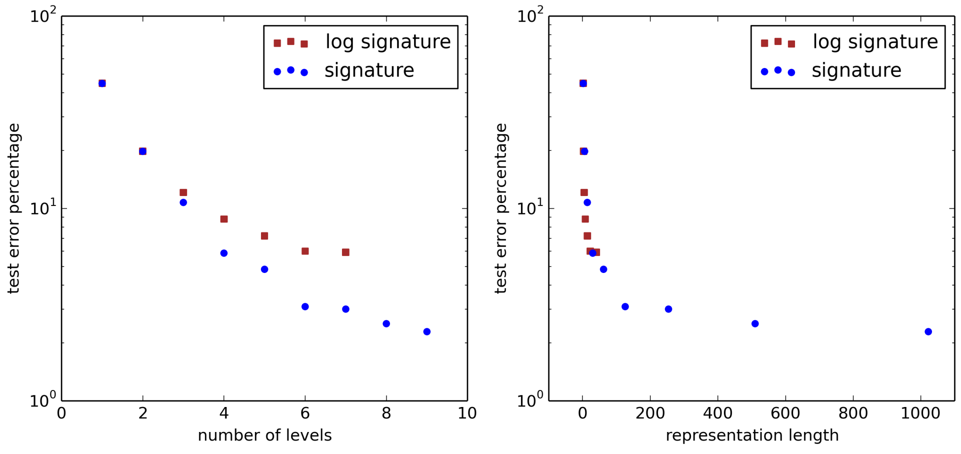

Figure 7: Test error (shown on a log scale) for different levels of signature and log signature

The log signature is more efficient for a given approximate representation length. The significant difference between five and six levels of log signature shows that there is useful information in the six-fold integrals.

Figure 8 compares the test error from using the signature with that from using linear rotational invariants or using the signature of the rotated path. The signature does always outperform the modified versions for each given level, so that the orientation is a useful property of the character. Hiding the orientation of the character from the algorithm does impair performance. The linear rotational invariants are very much shorter than the signature, and at a given representation length they are comparable with it in performance.

7.2 Extra dimension

We compared the effectiveness of the various representations which use a third dimension for strokes by trying them on the Assamese dataset. The Assamese dataset should be more challenging to our method of separating strokes because there are more strokes per character - up to 11, with an average of about 3. There are also more line segments per character (about 5 times more) which means that it takes longer to calculate signatures than for Pendigits.

The lengths of the representations used and the test error percentages found using each are given in Table 2. It is clear that using any of the extra dimensions enhances performance for a given level of signature. The results for each of the extra dimensions are plotted in Figure 9. The unnormalised ink dimension has the best performance. It makes sense that it does better than the normalised version, because it carries strictly more information.

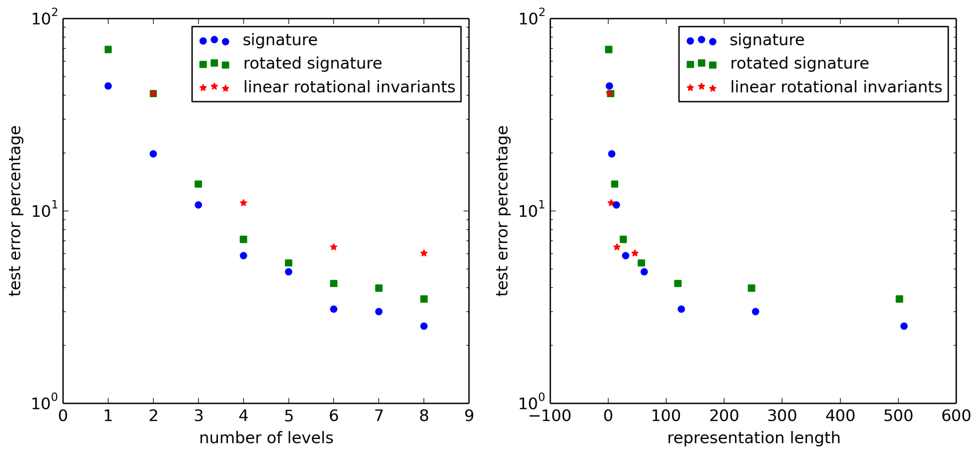

Figure 8: Test error (shown on a log scale) for different levels of signature and rotationally invariant versions

Levels included	2	3	4	5	6	7
Length of 2 dimensional signature	6	14	30	62	126	254
Test error with no extra dimension	82.64	70.98	69.28	63.33	60.11	56.53
Length of 3 dimensional signature	12	39	120			
Test error with ink dimension	67.82	50.7	39.04			
Test error with normalised ink dimension	70.73	57.32	43.53			
Test error with pen dimension	68.18	53.13	45.48			
Test error with stroke dimension	69.03	55.31	46.87			

Table 2: Representation length and minimum test error percentage for Assamese with various extra dimensions

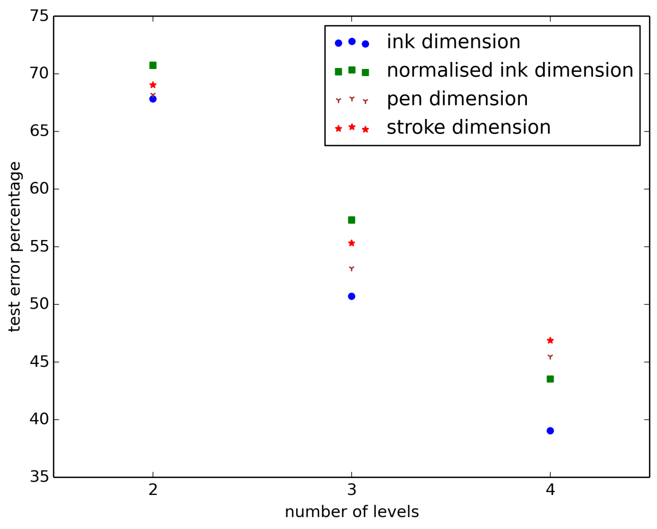

Figure 9: Test error for different levels of signature of Assamese characters, with either no extra dimension (just the concatenated character) or various possibilities for the third dimension

8 Conclusions and further directions

In seeking to improve representations of characters, we have seen that keeping information about the orientation of the character is useful. There is useful information in the iterated integrals which form the signature at least up to level six. The ink dimension is the best way to encode multiple-strokes in a three dimensional path.

In particular with Assamese, the high number of line segments per character (on average about 200) makes calculating high levels of signatures slow. We could consider ways to approximate each character by a smaller number of line segments without losing much of the shape of each character.

In this work we have looked at rotational invariance separately from the extra dimension. We have derived the form of relevant linear rotational invariants in three dimensions (see Appendix A). We would like to look at the effect of using linear rotational invariants together with the extra dimensions.

There are some small modifications to some of the representations we have considered which might be helpful. For example, we could add the angle of rotation as an extra element to one of the rotationally invariant representations, or the total ink used as an extra element to the representation using the normalised ink dimension. This lets the classifier decide whether the extra fact is important.

In this work, we have not compared our representations with the very different representations which are currently standard. For example the *histogram method* ([2]) in which the bounding box is divided into a square grid and the amount of ink spilled in moving the pen in each of a fixed set of directions in each square is an element of the representation. If we did so, we could consider whether and how we could enhance those other representations by adding data from some of ours.

Ultimately the best method of recognizing characters is usually considered to be the one which achieves the least test error for a given training time. We have compared representation functions for a fixed amount of training. This was thought reasonable because all the representations we have been considering should be able to be calculated quickly and in parallel and so most of the run time in an ideal implementation should be in training the neural network. However this is not always the case, with larger signatures being slower to calculate and this sometimes accounting for a significant proportion of the training time. Also, the comparison between representations of very different lengths is not quite fair, because the time taken by the network to train a batch depends on the size of the first layer of the network. It would be better to run all our tests for a fixed time of training rather than a fixed number of batches.

There are many classification algorithms, and some of our results could be dependent on the idiosyncrasies of our chosen one. Our neural network is a general-purpose classifier, not at all specially adapted to the problem at hand. There are adaptations which might be useful for some of these representations. For example, the stroke dimension is difficult for our neural network to learn from, but this could be adapted to.

Appendix A Considerations around linear rotational invariants

Linear rotational invariants are defined in Section 5. In two dimensions, a thorough explanation and derivation of them is given by [5]. We use the notation of the that paper, some of which is not defined above. Also, in section 4.2 there (pages 5–6), an attempt is made to explicitly present the linear rotational invariants up to level 6. The vectors given there (as $I_3, \dots, I_{15}$) for levels 4 and 6 do not appear to be rotationally invariant. There is no obviously best way to give a basis of the subspace of additional invariants in each dimension, so it is not clear exactly which vectors were intended.

For level 4, a basis of additional vectors is

$$\begin{aligned}
I_3' &= -x_1x_1x_2x_2 + 2x_1x_2x_1x_2 - x_1x_2x_2x_1 - x_2x_1x_1x_2 + 2x_2x_1x_2x_1 - x_2x_2x_1x_1 \\
I_4' &= x_1x_1x_1x_2 - x_1x_1x_2x_1 - x_1x_2x_1x_1 - x_1x_2x_2x_2 \\
&\quad + x_2x_1x_1x_1 + x_2x_1x_2x_2 + x_2x_2x_1x_2 - x_2x_2x_2x_1 \\
I_5' &= x_1x_1x_1x_2 - 3x_1x_1x_2x_1 + 3x_1x_2x_1x_1 + x_1x_2x_2x_2 \\
&\quad - x_2x_1x_1x_1 - 3x_2x_1x_2x_2 + 3x_2x_2x_1x_2 - x_2x_2x_2x_1.
\end{aligned}$$

The argument in the paper can be extended to determine those linear combinations of signature elements of a three dimensional path which are invariant under rotation in the x_1x_2-plane. Theorem 2 there becomes

Theorem 1 *Let $n \geq 2$ and $i_1, \dots, i_n \in \{1, 2, 3\}$ be such that*

$$\#\{k : i_k = 1\} = \#\{k : i_k = 2\}. \tag{4}$$

Then $\phi := c_{i_1 \dots i_n}$ is invariant under rotation in the x_1x_2-plane, where

$$\begin{aligned}
c_{i_1 \dots i_n} &:= z_{i_1} \cdot z_{i_2} \cdot \ldots \cdot z_{i_n} \\
z_1 &:= x_1 + ix_2 \\
z_2 &:= x_1 - ix_2 \\
z_3 &:= x_3.
\end{aligned}$$

This theorem provides all the invariants. Note that they are not restricted to even numbered levels. For example, up to level 3 these invariants are

$$\begin{aligned}
J_1 &= x_3 \\
J_2 &= x_1x_1 + x_2x_2 \\
J_3 &= x_1x_2 - x_2x_1 \\
J_4 &= x_3x_3
\end{aligned}
\qquad
\begin{aligned}
J_5 &= x_3x_1x_1 + x_3x_2x_2 \\
J_6 &= x_1x_3x_1 + x_2x_3x_2 \\
J_7 &= x_1x_1x_3 + x_2x_2x_3
\end{aligned}
\qquad
\begin{aligned}
J_8 &= x_3x_1x_2 - x_3x_2x_1 \\
J_9 &= x_1x_3x_2 - x_2x_3x_1 \\
J_{10} &= x_1x_2x_3 - x_2x_1x_3 \\
J_{11} &= x_3x_3x_3.
\end{aligned}$$

References

[1] E. Alpaydin and Fevzi. Alimoglu. *Pen-Based Recognition of Handwritten Digits Data Set.* 1998. URL: `https://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits` (cit. on p. 2).

[2] Zhen-Long Bai and Qiang Huo. "A study on the use of 8-directional features for online handwritten Chinese character recognition". In: *Eighth International Conference on Document Analysis and Recognition, 2005. Proceedings.* Aug. 2005, 262–266 Vol. 1 (cit. on p. 9).

[3] Udayan Baruah and Shyamanta M Hazarika. *Online Handwritten Assamese Characters Dataset Data Set.* 2011. URL: `https://archive.ics.uci.edu/ml/datasets/Online+Handwritten+Assamese+Characters+Dataset` (cit. on p. 2).

[4] Kuo-Tsai Chen. "Integration of Paths – A Faithful Representation of Paths by Noncommutative Formal Power Series". In: *Transactions of the American Mathematical Society* 89.2 (1958), pp. 395–407 (cit. on p. 3).

[5] Joscha Diehl. "Rotation invariants of two dimensional curves based on iterated integrals". 2013. URL: `http://arxiv.org/abs/1305.6883` (cit. on pp. 2, 4, 6, 9).

[6] Benjamin Graham. "Sparse arrays of signatures for online character recognition". 2013. URL: `http://arxiv.org/abs/1308.0371` (cit. on p. 2).

[7] Ben Hambly and Terry Lyons. "Uniqueness for the signature of a path of bounded variation and the reduced path group". 2005. URL: http://arxiv.org/abs/math/0507536 (cit. on p. 3).

[8] Terry Lyons. *System control and rough paths.* 2002 (cit. on p. 2).

[9] Terry Lyons et al. *CoRoPa Computational Rough Paths (software library).* 2010. URL: http://coropa.sourceforge.net/ (cit. on p. 4).

[10] Christophe Reutenauer. *Free Lie Algebras.* 1994 (cit. on p. 4).