Neural Networks for Pen Characters Recognition

Carlos Agell
Department of Electrical Engineering and Computer Science
University of California, Irvine
Irvine, CA 96217
cagell@uci.edu

Abstract

This paper presents experiments that compare different kinds of features for isolated handwritten character recognition in a Neural Network environtment. The selection of valuable features is crucial in character recognition, therefore a new and meaningful set of features, the Uniform Differential Normalized Coordinates (UDNC), is introduced. These features are shown to improve the recognition rate using simple classification algorithms so they are used to train a Neural Network and test its performance on a reduced database.

1 Introduction

Character recognition, both human and machine generated, is a wide and largely studied field in Machine Learning. In fact, nowadays many commercial scanners use *Optical Character Recognition* (OCR) systems that output a character string having an image of typed text as input.

In the age of Personal Digital Assistants (PDA) and tablet PCs, there is an upgrowing need and interest for handwritten character recognition. This kind of recognition, slightly different from the general OCR approach, will be the scope of this paper.

This paper proposes to solve the classification of isolated handwritten characters and digits of the *UJI Pen Characters Data Set*[1] using Neural Networks. The data [6] consists of samples of 26 characters and 10 digits written by 11 writers on a tablet PC. The characters (in standard UNIPEN format[1]) are written both in upper and lower case and there is a whole two set of characters per writer. So the output should be in one of the 35 classes. The ultimate objective is building a writer independent model for each character.

The approach of this paper leans in two main points: extracting meaningful features and training a neural network to correctly classify the data.

The rest of the paper is organized as follows: after this introduction there is a brief review of the state of the art in character recognition and the different approaches taken to solve this problem. Section 3 is dedicated to select the appropriate features, where a new set

[1] See the UNIPEN Project history in http://hwr.nici.kun.nl/unipen/

of features for standard UNIPEN characters is introduced. Following there is a section describing the implementation the approach in this paper. Section 5 shows the analysis of the results from the implementation. Finally the conclusion points out the main novelties reported in this paper.

2 Previous Work

Several approaches might be found in literature to solve the character recognition problem using different Machine Learning techniques and algorithms. This paper will only discuss about isolated character recognition.

On the one hand the typical character recognition has been attacked with tools such as simple Nearest Neighbour[4], Support Vector Machines (SVM)[8] and Neural Networks[3, 4], that leaded to impressive results on different databases. But this approaches are all used when the input are images, such as the classical MNIST database of mail zip codes [5].

On the other hand, for the specific case of UNIPEN character recognition, there are diverse approaches feature-wise and algorithm-wise.

Regional-Fuzzy Representation (RFR) lead to impressive recognition rates when using them as inputs for a Neural Network [7]. The experiment framework for this case, though, worked only with lower-case, upper-case and digits separately.

A whole discussion about feature selection can be found in [2], where part of the work of this project was inspired on.

3 Feature Selection

A correct feature selection is crucial in any pattern recognition problem and handwritten character recognition is not an exception.

If we want to classify we need to measure in dimensional space common for all the characters (recall that the number of coordinates in a UNIPEN representation varies depending on the character).

This section describes the two main approaches proposed as features for the UJI database: making images from the features and processing the UNIPEN data to make meaningful features. We also introduce the Uniform Differential Normalized Coordinates (UDNC) as part of the second approach. Finally, we compare all the features using a simple classification algorithm.

3.1 Making Images from the Features

Making images out of the UNIPEN coordinates seems a correct approach for character recognition as there is a strong background in this problem using those kind of features.

In order to get feature images to use for recognition we first center the data and linearly interpolate on a grid so that we can reconstruct the original character.

Computationallywise we would like to have the fewer dimensions as possible so that the classification process does not take long, that is why we could think about decimation. But decimation makes no sense in a linearly interpolated image because the connections between points will be lost. For this reason we think it is useful to use mathematical morphology, to erode the initial image and then decimate it. This way we can euristically lower the dimensions of the image, up to a representation of 30×40 pixel image which seems reasonally good.

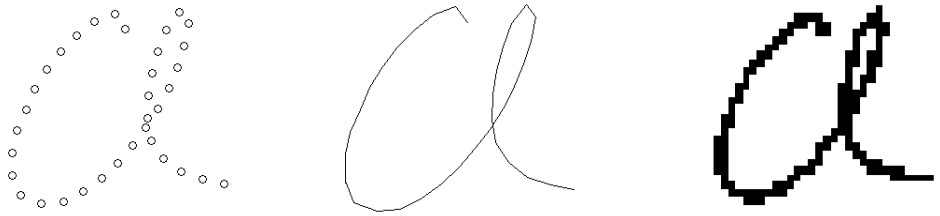

Figure 1: Coordinates of the UNIPEN format data (left), linear interpolation (center), and final 30×40 pixel image

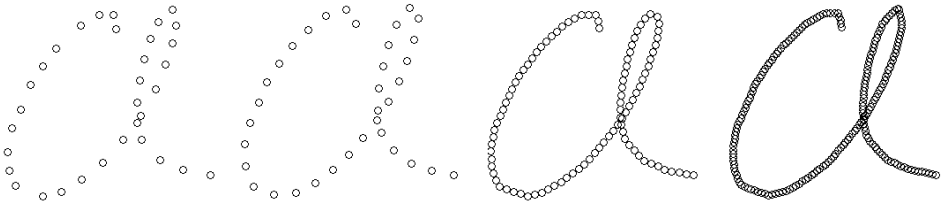

Figure 2: Coordinates of the UNIPEN format data (left) and uniform sampling using 35,100 and 200(right)

The initial coordinates, the linear interpolation and the final 30×40 pixel image are depicted in figure 1.

However there are several drawbacks in the use of these kind of features. First of all they are *not* actually for online character recognition as they need to be converted to images for further classification. This means postprocessing and not *online* processing as it is pointed out in the goal of the UJI database [6]. And secondly we are missing an important piece of information when converting to images: there is a time information hidden (some coordinates are generated first than others).

3.2 UNIPEN Feature Processing

With the improvement of the features in the previous section as a goal, this section offers different approaches to the features to use for the UNIPEN handwritten character recognition problem. Some ideas are introduced to face the problems UNIPEN format imposes. The section culminates with the definition of the Uniform Differential Normalized Coordinates which will overcome all the UNIPEN format problems.

3.2.1 UNIPEN format drawbacks

The main drawback of the UNIPEN format is that is not a single dimensional format for all the data. Each datapoint has different number of coordinates.

The previous section faced this problem by expanding the number of dimensions but somehow, this approach introduces redundant and non useful data.

We propose a **uniform sampling** for the UNIPEN format. This means to get the same number of coordinates for all the characters by resampling them using linear interpolation. Figure 2 shows this idea.

Once we have the dimensionality problem solved, we need to overcome another one, position and rotation (maybe the character is slighly in italics). This fact is also present in the image features, though is slightly overcome by centering. The problem could be solved using **relative coordinates** instead of absolute ones.

Finally there is another drawback, the length of each character is different. This means that when if we uniformly resample we are not taking into account the total lenght of the character (sum of the distances between all the points). **Normalization** could address this fact.

The three improvements cited are putted together in the Uniform Differential Normalized Coordinates.

3.2.2 Uniform Differential Normalized Coordinates

The Uniform Differential Normalized Coordinates (UDNC) are defined as the difference between the uniformly resampled coordinates normalized by the length of the character.

The construction of this kind of coordinates follows the following algorithm, assuming we have the original UNIPEN format data.

- Uniformly resample each character using linear interpolation ($U_i, U \in \Re^{n+2\times 2}$).
- Calculate the difference between datapoints ($UD_i = U_{i+1} - U_i, D \in \Re^{n\times 2}$).
- Normalize by the total length of the character, $UDN_i = \frac{UD_i}{\sum_i ||UD_i||}, UDN \in \Re^{n\times 2}$

It must be pointed out that some of the characters have multiple traces (to write an X you need to put the pen up from the paper). Those characters are uniformly resampled proportionally to the length of each of the traces.

The parameter n can be choosen, so the final dimensions will be $2n$. For this work and the particular case of the characters the value $n = 35$ allowed a correct reconstruction of all the characters.

These UDC coordinates are normalized, invariant to rotations and displacements, and have the same dimensions to represent all the data. Furthermore they can be applied to recognize handwritten characters *online*.

3.3 Feature Comparison

As training a Neural Network can be computationally expensive, this section exposes the comparison of different features using simple classification algorithms. These simple classification algorithms are computationally reasonable and allow crossvalidation to see which features performs better. The best behaving features will further be used by the Neural Network implementation to get a better classification.

Table 1 shows the performance of the features tried in this paper for leave-one-out crossvalidation. The simple algorithms applied are mainly Nearest Neighbor [4], Nearest Mean and its SVD equivalent.

Nearest Mean is nothing more than choosing the closest mean to the input data, when the mean has been calculated using all the data except for the characters of a particular writer. The SVD equivalent is nothing more than using the projections of the data on the first k eigenvectors and using them as features. This last algorithm used $k = 100$ and was an approach to reduce dimensionality.

Table 1: Recognition Rates for different algorithms and features

Features	Algorithm	Max	Mean
Image 30×40	1-Nearest Neighbour	45.97%	31.82%
Image 30×40	1-Nearest Mean	62.19%	43.84%
Image 30×40	SVD 1-Nearest Mean	53.23%	40.32%
Uniform Coordinates($n = 35$)	1-Nearest Neighbour	77.42%	63.56%
Differential Angles($n = 35$)	1-Nearest Neighbour	69.35%	55.65%
Uniform Differential($n = 35$)	1-Nearest Neighbour	83.06%	67.08%
UDN Coordinates($n = 35$)	1-Nearest Neighbour	83.87%	70.60%

4 Neural Network Implementation

This paper proposes a softmax neural net with 1 hidden layer and a fixed number of nodes (62) in the single hidden layer. The input features are 70 dimensional.

The original problem asks to classify in 35 groups, corresponding to the number of letters plus the number of digits except for the 0 (which is considered to be the same as 'O' and 'o'). Knowing that the data contains both upper and lower case letters, the approach of this paper proposes training a 62 output neural network with a posterior lookup table relating upper and lower case characters. This means that the Neural Network should be trained to distinguish the 62 different characters (62 outputs).

We will use the multi-class cross entropy error function defined in equations 1 and 2, with a bias term for both the hidden and the output node. The Neural Network will be trained using a classic backpropagation algorithm.

$$J(w) = -\sum_{i=1}^{m}\sum_{k=1}^{3} y_k^{(i)} log\left(out_k(x^{(i)})\right) \tag{1}$$

$$out_k(x^{(i)}) = \frac{exp(a_{out,k})}{\sum_j exp(a_{out,j})} \tag{2}$$

The minimization in the backpropagation algorithm implemented uses a 5-level depth 9-points long line search.

For timing constraints the neural network was trained for a limited amount of iterations (15 iterations for each of the datapoints) and not until convergence.

5 Complete System

Figure 3 shows a block diagram of the different steps in the recognition scheme.

6 Results

In this section we present the results and a short analysis of the error committed.

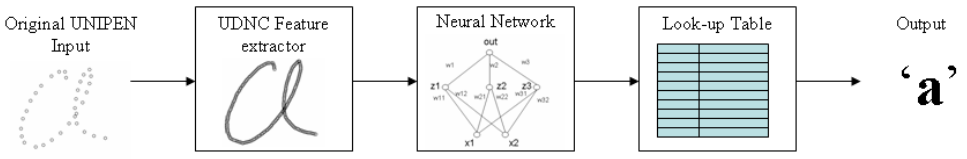

Figure 3: Block diagram of the complete system

Table 2: Algorithm comparison for Cross-Validation and UDNC Features

Algorithm	Max Recognition Rate	Mean Recognition Rate
1-Nearest Neighbour	71.57%	67.71%
1-Layer Neural Network	75.81%	72.99%

6.1 Performance

The results reported are related to considering 7 writers in the database as training and the rest (4) as testing. We performed 4 cross-validations (taking 4 different sets of 7 writers as training). Table 2 shows the performance of the Neural Network implementation versus the 1-Nearest Neighbour (baseline).

6.2 Error Analysis

In order to see where does the Neural Network commit mistakes we build a confusion matrix as we can see in figure 4. The values of this matrix correspond to the best performing classification using Neural Networks.

It is funny to see that there are several errors, specially confusing characters that, indeed, look alike. The most confused pairs can easily be found looking for the maximum of the confusion matrix. Those are the following: `a-q`, `9-q`, `r-t`, `r-y` and `2-z`.

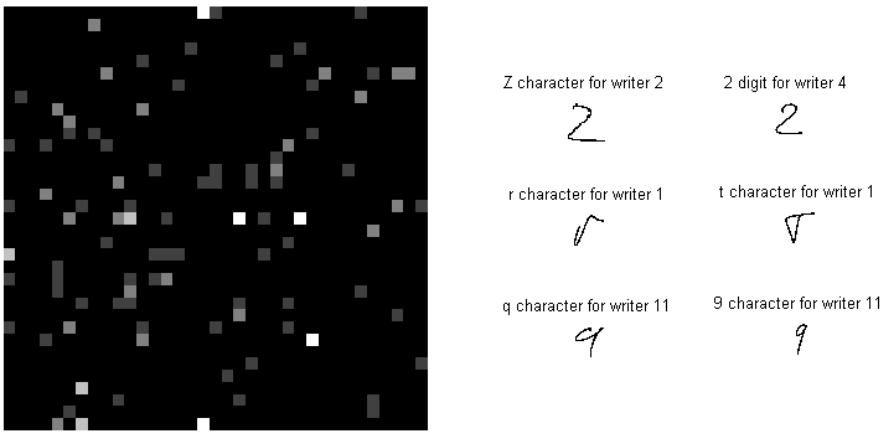

Figure 4: Confusion Matrix and a pairs of values where the Neural Network does not correctly recognize.

7 Conclusion and Future Work

In this paper we have reviewed different kinds of features to recognize handwritten characters and proposed novel approach: the Uniform Differential Normalized Coordinates. These features outperform regular features for character recognition.

The Uniform Differential Normalized Coordinates have been used to recognize the characters from a public database using standard Neural Networks. The results improve the baseline of the 1-Nearest Neighbour algorithm.

Several other variants of this work could be reviewed more deeply. Future work on this project would include cross validation to check which is the optimal number of hidden neurons, comparison with other kinds of algorithms and iteration of neural networks until convergence instead of a certain number of iterations.

Acknowledgments

This work has been funded by the program of graduate fellowships of the *Departamento de Becas, Investigacion y Universidades* of *Fundacion Caja Madrid* .

References

[1] A. Asuncion and D. Newman. Uci machine learning repository. `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 2007.

[2] C. Bahlmann. Directional features in online handwriting recognition. *Pattern Recognition*, 39(1):115–125, 2006.

[3] M. Garris, C. Wilson, and J. Blue. Neural network-based systems for handprint ocr applications. *IEEE Trans. Image Processing*, (7):1097–1112, 1998.

[4] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[5] Y. LeCun and C. Cortes. The mnist database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

[6] D. Llorens, F. Prat, A. Marzal, and J. M. Vilar. The uji pen characters data set. `http://archive.ics.uci.edu/ml/datasets/UJI+Pen+Characters`.

[7] M. Parizeau, A. Lemieux, and C. Gagné. Character recognition experiments using unipen data. *Proc. of 6th Int. Conf. on Document Analysis and Recognition (ICDAR)*, pages 481–485, 2001.

[8] J. xiong Dong, A. Krzyzak, and C. Y. Suen. An improved handwritten chinese character recognition system using support vector machine. *Pattern Recogn. Lett.*, 26(12):1849–1856, 2005.