Applying Domain Knowledge to the Recognition of Handwritten ZIP Codes

Ibrahim Y. Chaaban
Indiana University at South Bend

ABSTRACT

We present a simple system that exploits domain knowledge to improve the segmentation and recognition of handwritten ZIP codes. Specifically, we show that the concept of metaclasses of digits, introduced by Morita et al. [16] for recognition of Brazilian bank check dates, can be extended to ZIP code recognition. We also show that, when this domain knowledge is present, integrated segmentation and recognition is not required for the recognition of handwritten ZIP codes, as claimed by Liu et al. [4].

INTRODUCTION

Machine recognition of handwritten digits is very challenging. The difficulties and complexity of this task lie in the fact that a system must be able to recognize handwritten digits produced by different people, using different instruments. A system has to deal with widely different sizes and slants, as well as different shapes and widths of the strokes, for example. Accordingly, many approaches and methods have been proposed for pre-processing, feature extraction, classification and/or learning of handwritten digit images.

Handwritten digit recognition research concentrates on either individual digits or digit strings. With respect to recognition of individual handwritten digits, machine vision has achieved an accuracy of 99.6% [17]. By comparison, humans recognize 97.63% of individual handwritten digits [10]. Many domains require recognition of digit strings, as opposed to individual digits however. Some examples are automated sorting of mail by postal code [7], automated reading of checks [15] and tax returns, and data entry for hand-held computers. In these domains, handwritten digits rarely appear isolated. Instead they appear as part of a string of digits where some digits may touch and/or overlap. In many of these real world applications, the images are processed by human operators. However, automation may improve production and cut costs. For this to happen, performance of an automated system should compare favorably to human performance. Such comparison is also an essential component in determining whether the problem has been solved or not.

In the domain of bank check processing, Morita and colleagues [16] developed a system that uses the Hidden Markov Model (HMM) and the Multilayer Perceptron (MLP) to segment and recognize unconstrained handwritten dates on Brazilian bank checks. The system processes the three subfields that make up the date (day, month, and year). In order to reduce the date lexicon size, Morita and colleagues used domain knowledge, which enabled them to reduce the complexity of the recognition process. This was possible because the lexicons for day and year are known. For example, in a two digit day the first digit can only be 0, 1, 2, or 3, while the second digit can range from 0 to 9. A similar approach was applied to two/four digit years. They restricted checks to just those written after 1990 and before 2029. In a two digit year the first digit can then be 0, 1, 2, or 9. The second digit can range from 0 to 9. In a four digit year, the first digit can only be 1 or 2; similarly the

second digit can only be 0 or 9. They used five MLP neural networks, each using one hidden layer, to classify the five different metaclasses of digits. For example, one MLP network classified the digits in the metaclass consisting of 0, 1, 2 and 3. It had one hidden layer with 70 hidden units. Another network classified the digits 0 through 9 using one hidden layer with 80 hidden units, and so forth. The hidden layer in each network contained an empirically determined number of hidden units, each of which connected to all input and output units. The performance of this system on four digit years was 100%. On two digit days performance was 93.2%, and on two digit years performance was 97.2%. The discrepancy between performance on two digit years and two digit days can also be explained by their use of domain knowledge. Specifically, they exploited the fact that year always appears at the end of a date. They used this knowledge to improve segmentation of the year from the rest of the date, and the improved segmentation led to improved recognition.

ZIP code recognition is another very interesting problem, due to the benefits of having an accurate automated system that can sort letters at a high rate. On average a postal worker can sort about 800 letters an hour. On the other hand, an automated sorting machine, reading printed ZIP codes with an optical scanner is estimated to process about 37 times more than the postal worker at a fraction of the cost [13]. Such performance would also be desirable for handwritten ZIP codes. Liu and colleagues [4] compared different classifiers and learning methods in the recognition of handwritten ZIP codes. The classifiers compared were single-layer perceptron (SLP), multi-layer perceptron (MLP), radial basis function classifier (RBF), polynomial classifier (PC), learning vector quantization (LVQ), modified quadratic discriminant function (MQDF), and learning quadratic discriminant function (LQDF). Each classifier had two or three variations depending on the learning method, such as maximum likelihood estimation (MLE), discriminative learning (DL), or enhanced discriminative learning (EDL). The method of maximum likelihood is a general method of estimating parameters of a population by values that maximize the likelihood of a sample [5]. The discriminative learning method on the other hand, updates parameters iteratively to separate the patterns of different classes. The enhanced version of discriminative learning is equivalent to DL, except that in EDL the training is done with outliers [4]. The first classifier they tested was the SLP. A single layer perceptron has an input and output layer. Each neuron in the output layer of their network was connected to each input neuron. When trained with the EDL method and forced to make a decision without rejection, this network's correct recognition rate was 74.31%. This is probably because this type of network is limited to only a single layer. They also tested a MLP network. With one or two hidden layers, this network can approximate virtually any input-output map, by learning to transform input data into a desired response. The MLP's correct recognition was 89.22% without rejection using the same learning method. When using the EDL method, both the RBF and PC produced similar results to that of the MLP. The RBF correct rate without rejection was 87.84%, and the PC had a correct rate of 89.91% without rejection. The LVQ classifier was also tested. LVQ is a competitive learning algorithm, described sometimes as the supervised version of Kohonen's Self-Organizing Map [19]. Percent correct for the LVQ classifier was 87.61% with no rejections. The MQDF classifier, described by Liu et al. [4] as the MLE version of LQDF, was also tested, and it had a correct rate of 87.61% without rejection. Finally, the LQDF classifier was tested to reveal a 90.37% correct rate without rejection.

In order to test these classification methods, Liu and colleagues [4] developed a new model. First, the model pre-processed the string image to prepare it for pre-segmentation. In the pre-segmentation stage, connected component labeling was applied. To handle the cases of touching digits, the model analyzed the upper and lower profile curves of any touching digits in order to generate a candidate cut. Heuristic rules were also applied to ensure that the candidate cut would not split a single digit. Their pre-segmentation stage was followed by an integrated segmentation and recognition (ISR) stage. In this stage Liu et al. [4] combined dynamic programming (DP) search and digit recognition. Each of the character classifiers described earlier was used to assign class scores to the candidate patterns generated in the pre-segmentation stage. The optimal pattern was then found by DP search based on class scores given by the classifiers. According to Liu and colleagues [4], the digits in a ZIP code cannot be reliably segmented in a distinct stage prior to recognition. They claimed that the handwritten ZIP code problem cannot be solved without *integrated segmentation and recognition*. Integrated segmentation and recognition was used by many researchers in different areas (e.g. Xin and colleagues [20] used ISR to recognize characters on license plates; Liu et al. [6] also showed how ISR can be used in the recognition of numeral strings). The recognition portion of this integrated segmentation and recognition stage tested the different types of classification methods described above. The LQDF classifier, as noted, had the best performance. The system was tested on 436 5-digit ZIP code images from CEDAR CDROM-1 [1]. The ZIP code images were obtained by USPS from actual mail images. Liu and colleagues [4] reported a correct recognition rate of 90.37%, which appears to be the best performance to date in machine recognition of handwritten ZIP codes. By comparison, humans recognize 98.39% of handwritten ZIP codes [11].

Unlike the Brazilian bank check date recognition system of Morita et al. [16], domain knowledge was not used in the U.S. ZIP code recognition system of Liu et al [4]. Might the metaclasses technique of Morita et al. [16] generalize from the case of Brazilian bank check dates to the case of U.S. ZIP codes? Furthermore, if such a domain knowledge technique were applied to automated ZIP code recognition, would integrated segmentation and recognition be required, as claimed by Liu et al. [4], or might a distinct segmentation stage followed by a distinct recognition stage work just as well? Our work addresses both questions.

MODEL

A key feature of our model is that it incorporates domain knowledge. The structure of ZIP codes and the state of destination can be exploited to make the segmentation and recognition processes more accurate. Understanding the structure of ZIP codes can significantly reduce the range of possible classes to consider during the recognition process, thereby increasing accuracy. For example, in a five digit ZIP code, the first digit indicates one of ten large geographic areas in the country. It represents a certain group of U.S. states, ranging from zero in the Northeast to nine in the far West. The second and third digits indicate metropolitan areas and sectional centers. The fourth and fifth digits represent more specific areas such as local post offices or postal zones in larger cities [22]. Combining this knowledge with the state of destination radically reduces the number of ZIP codes to consider during recognition.

Our model consists of two stages: a segmentation stage and a recognition stage. The segmentation stage exploits the fact that a ZIP code is composed of 5

digits[1]. Utilizing the destination state, the recognition stage tries to recognize the five segments derived from the segmentation stage, as shown in Figure 1.

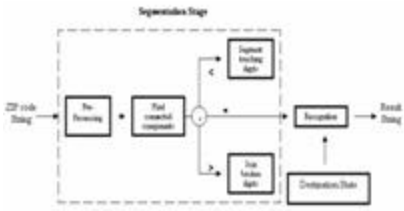

Figure 1. The two stages (segmentation & recognition) of our model.

Before describing our model and its performance, we make explicit certain assumptions, as well as our predictions. With respect to domain knowledge, as mentioned above, our model assumes five-digit ZIP codes. Also knowledge of the structure of ZIP codes (as briefly described above) is built into the model. Further, our model assumes perfect knowledge of the state of destination as an input. This latter assumption requires some justification. First, in the real world, destination state would come just before ZIP code. (Alternatively, it may appear on the line just above the ZIP code line.) Given that English is normally read from left to right and from top to bottom, it is reasonable to assume that humans would have knowledge of destination state when trying to read a ZIP code. We want to be able to compare our model's performance to human performance. Therefore, in order to make this comparison fair, destination state is input to our model for each ZIP code to be recognized. Second, the main purpose of this study is to address the two questions described in the introduction. Will the metaclasses technique of Morita et al. [16] generalize beyond Brazilian bank check dates to U.S ZIP codes? Does machine recognition of handwritten ZIP codes require an integrated segmentation and recognition stage, as claimed by Liu et al. [4], when domain knowledge is present? To decisively answer each question, knowledge of correct destination state for each ZIP code must be assumed by the model[2].

Segmentation

One challenge of classifying handwritten ZIP codes is the fact that in real applications the image extracted from a piece of mail will not necessarily appear as five separated digits. This is due to imperfect handwriting, as shown in Figure 2.

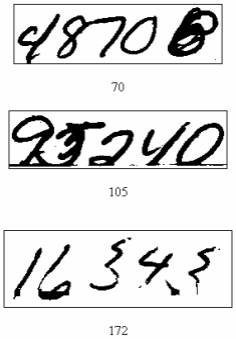

Figure 2. Sample patterns from the CEDAR CDROM-1 database.

In addition, extra noise (such as bar codes, stamps and other markings made by the post office) is added to the image during processing. Further, moisture and handling may smear or smudge the handwriting on an envelope. To overcome these challenges and to achieve segmentation, a

[1] We focused on five-digit ZIP codes vs. nine-digit ZIP codes for the following reasons: (1) Nine-digit ZIP codes are optional. (2) The testing data found on the standard CEDAR CDROM-1 did not have enough nine-digit ZIP codes for testing. (3) Liu and colleagues [4] used five-digit ZIP codes in testing their system, and so did we to allow for seamless comparison with their work.

[2] While our goal was not to develop our model for real-world deployment, it could easily be modified for this purpose by taking into account a relaxation of the assumption of correct destination state for a given ZIP code.

ZIP code pattern goes through three phases in our model (see Figure 1).

In the first phase, the ZIP code pattern is prepared for division into five separate patterns, each representing a digit in the ZIP code. First the pattern is converted into a binary image. This allows for easy processing of patterns. Then, from the pattern a median filter removes any set of connected pixels with an area-size less than an empirically determined threshold (3×3). This step is necessary because it helps to remove some of the added noise discussed earlier. Next, the image closing algorithm is applied to the pattern to perform morphological "closing" on the binary digits found in the image (see Figure 3) [2]. This step will enhance the shape of each digit and therefore allow for better classification later in the process [18].

Figure 3. Sample patterns extracted from ZIP codes found on the CEDAR CDROM-1 database. The patterns show how some digits are incomplete because of gaps.

The second phase of segmentation finds all connected components in a given ZIP code pattern. A connected component is a set of pixels sharing some feature (blackness of the pixel) where each pixel in the set neighbors at least one other pixel in the set. The purpose of this step is to isolate the digits which make up the ZIP code pattern. The connected components algorithm checks four neighboring pixels to determine connectivity to other pixels [14].

Because a ZIP code is composed of 5 digits, the second phase can have three possible outcomes. The first outcome would be to get five connected components with a relatively similar size. This would imply that the pattern was successfully segmented into five digits. The second outcome would be to get less than five connected components. This could mean that two or more digits in the ZIP code are touching or overlapping. If so, they must be separated. The third outcome would be to get more than five connected components. This particular outcome would suggest that there may be one or more individual digits that are broken into multiple pieces which therefore must be joined together.

The third phase of segmentation involves separating connected digits (if necessary) or joining multiple segments of a digit (if necessary) found in a ZIP code.

Separating touching digits. When two or more digits are touching or overlapping, separating them is particularly challenging. An improper separation of these digits tends to leave some of the newly separated digits with noise or parts of the touching digit(s). In the separation process a digit might lose a piece of a stroke to a neighboring digit, or a digit might lose a chunk because of overlapping. These problems arise because finding the precise splitting path that separates touching digits is nontrivial (see Figure 4).

Figure 4. A sample of ZIP codes from CEDAR CDROM-1. The samples show that finding the precise splitting path is nontrivial.

To split touching digits our algorithm starts by finding the bounding

box of the entire ZIP code in a given pattern. Then the algorithm divides the area of the bounding box vertically into five relatively equal segments. This algorithm tends to leave segments with noise or parts of the neighboring digit(s). In order to filter out the noise found in each of the five segments, the largest component is located in each segment and everything else (noise or parts of neighboring digits) is discarded. The result of this algorithm is five relatively noise-free segments each representing a single isolated digit ready to be converted into a feature vector. The process of converting a ZIP code pattern into a vector is described in the Feature Extraction section.

Joining segments of a digit. Joining segments that belong to one digit is as challenging as separating touching digits. The challenges here are to determine which segment belongs to which digit and whether a segment is part of a digit or simply noise. Furthermore, using the improper joining algorithm may leave some of the digits with artifacts or noise. These problems arise because finding the precise technique to join segments is nontrivial. A digit may occur in pieces as a result of three things: not having the proper writing tool (dry pen), noise or extra markings added during processing, or the writer did not properly connect the digit. By analyzing various cases of broken digits, we found that the digit which often occurs in two segments is the digit 5. With the exception of the digit 4, the digit 5 is the only digit which often is written in two parts (see Figure 5). Therefore, the digit 5 is perhaps more likely to appear in two parts than any of the other digits.

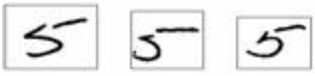

Figure 5. Sample patterns extracted from ZIP codes found on the CEDAR CDROM-1 database. The patterns show how the digit 5 is sometimes presented in two segments.

To solve this problem we used the single line test algorithm. This algorithm is commonly used in mathematics [12]. It tests columns of pixels one at a time as it moves across the entire ZIP code starting from the left-hand-side. If the column being tested intersects the area of two of the connected components found earlier, then those two components are joined. In case of a failure to find two components to connect, the ZIP code is simply divided into five equal segments as described in the previous section on separating touching digits.

Feature Extraction

Since different individuals can have various writing styles, the features extracted from each digit must be independent of size, width of the strokes, and other elements of the writing styles of the individuals. To attain such a feature vector a matrix of pixels is first sampled from each digit pattern, as shown in Figure 6.

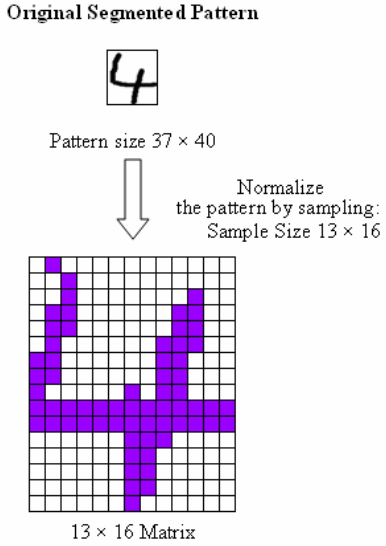

Figure 6. Normalizing the segmented digits by sampling.

Determining the proper sample size means extracting meaningful features using the smallest sample size possible. Careful evaluation of the size, width of the strokes, and the shapes of the training patterns indicated that a sample size of 208 pixels (13 × 16) would provide good representation of the patterns[3]. Then, the matrix is reduced to a vector by projecting it onto the X-axis, as shown in Figure 7.

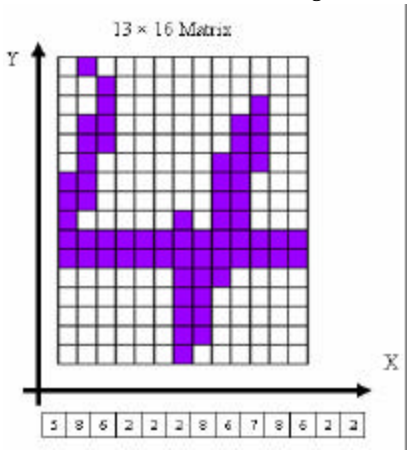

Figure 7. Converting the 13 × 16 matrix into a vector by projecting onto the X-axis.

The resulting feature vector contains values that represent the number of black pixels found in each column of the matrix. A similar technique was used by Rababaah [9] to reduce the size of asphalt pavement crack patterns. To illustrate the uniqueness of these vectors we created two graphs. These graphs show that despite the resemblance in shape between the digit 0 and 8, their vectors are clearly unique, see Figure 8.

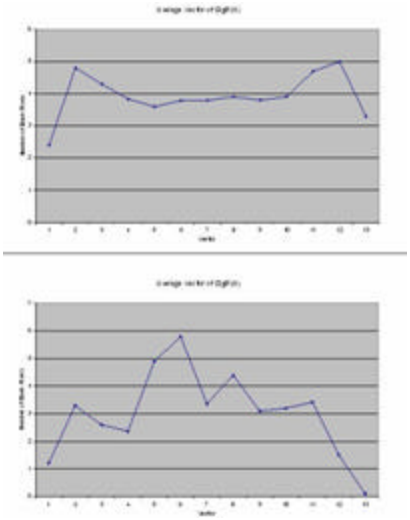

Figure 8. The top graph shows the average of 10 vectors each representing the '0' pattern. The bottom graph shows the average of 10 vectors each representing the '8' pattern.

Recognition

The task of the recognition stage is to recognize the string of individually segmented digits. Utilizing the destination state, the recognition stage tries to recognize each digit starting with the left-most digit, as shown in Figure 9.

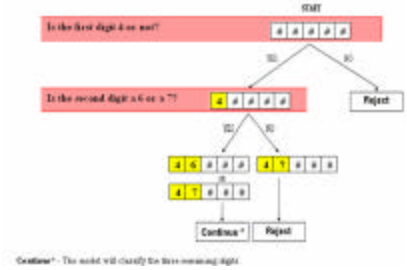

Figure 9. The knowledge-based recognition model starts with five unknown segmented digits. The model will first attempt to classify the left-most digit. If unsuccessful the ZIP code is

[3] For some small extracted digits, the 13 × 16 sample size presented a problem. Because the sample size was greater than the size of the digit pattern in these cases, there were not enough pixels in the pattern from which to sample. To solve this problem we used all the pixels in the pattern and completed the rest of the vector with zeros.

rejected, otherwise it will go on to classify the second digit. If the second digit is classified correctly the model will classify the remaining three digits, otherwise it will stop. (Indiana is the state assumed in this example.)

Because destination state determines the first two digits of a ZIP code, the system will make a decision for each of the first two digits on whether it is feasible to continue or not. Specifically, the metaclasses technique of Morita et al. [16] will be applied to each of the first two ZIP code digits. For example, if the destination state is Indiana, then the first digit must be a 4, because all ZIP codes in the state of Indiana start with the digit 4. Thus, the metaclass for the first digit of an Indiana ZIP code is {4}. Before going any further, the model must verify that the first digit is a 4, as shown at the top of Figure 9. If it is not, the model will stop and reject the ZIP code. On the other hand, if the first digit is a 4, then the model will proceed to classify the next digit. In this example, the model now must determine if the second digit is a 6 or 7, because in the state of Indiana these are the only possible digits after the initial '4'. Thus, the metaclass for the second digit of an Indiana ZIP code is {6, 7}. If the model classifies the second digit as a 6 or 7, then it will classify the remaining three digits. Otherwise it will stop and reject the ZIP code, as shown in Figure 9[4]. Recall that Morita and colleagues [16] used a similar strategy to classify handwritten dates (days/years) using domain knowledge. They were able to reduce the complexity of the recognition process by reducing the date lexicon size. This was possible because they knew the lexicons for day and year. For example, in a two digit day the first digit can only be 0, 1, 2 or 3 while the second digit can range from 0 to 9. A similar approach was applied to two and four digit years. For instance, in a two digit year, they only allowed the first digit to be 0, 1, 2 or 9. The second digit can range from 0 to 9. In their model, they identified five different metaclasses of digits: one metaclass for digits {0, 1, 2, and 3}, one for digits {0, 1, 2, 9}, one for digits {1, 2}, one for digits {0, 9}, and one for digits {0 through 9}. We conjectured that the metaclass approach employed by Morita et al. [16] may also be useful in recognition of U.S. ZIP codes. Obviously, the set of metaclasses useful for handwritten ZIP code recognition will be different than the set of five metaclasses useful for recognition of Brazilian bank check dates, Morita et al. [16]. The necessary set of metaclasses for handwritten ZIP code recognition will now be described.

[4] Here, in order to allow for seamless comparison to the results obtained by Liu et al. [4] rejection does not actually stop the processing of a given ZIP code. Instead the model passes the digits yet to be classified to another network which classifies each as one of the ten possible digits.

Classification Method

The recognition portion of this model was implemented using multi-layer perceptron (MLP) neural networks. Each individual network classified a distinct metaclass of digits. This was inspired by the use of distinct MLP classifiers for distinct metaclasses of digits in the study of Morita et al. [16]. One of ten MLP networks was used to classify the first digit of a ZIP code, depending on the state. Recall that in a five digit ZIP code the first digit corresponds to one of ten large geographic areas in the country. Therefore, we used ten networks (ten distinct metaclasses) to classify the first digit in a ZIP code. One MLP network classified the digits 6 and 7 (metaclass {6, 7}) for the second digit for the state of Indiana. Another classified the digits 0, 1, and 2 (metaclass {0, 1, 2}) for the second digit for the state of Illinois, and so forth for other states. This strategy required a total of twenty-six different MLP networks. Also, by exploiting the structure and range of possible ZIP codes for each state, we were able to create a list of possible second digit(s) for each state, as shown in Table 1.

States (ZIP code starts with 0)		Possible Second digit
CT	Connecticut	6
MA	Massachusetts	1 - 2
ME	Maine	3 - 4
NH	New Hampshire	3
NJ	New Jersey	7 - 8
RI	Rhode Island	2
VT	Vermont	5

States (ZIP code starts with 1)		Possible Second digit
DE	Delaware	9
NY	New York	0 - 1 - 2 - 3 - 4
PA	Pennsylvania	5 - 6 - 7 - 8 - 9

States (ZIP code starts with 2)		Possible Second digit
DC	District of Columbia	0
MD	Maryland	0 - 1
NC	North Carolina	7 - 8
SC	South Carolina	9
VA	Virginia	0 - 2 - 3 - 4
WV	West Virginia	4 - 5 - 6

States (ZIP code starts with 3)		Possible Second digit
AL	Alabama	5 - 6
FL	Florida	2 - 3 - 4
GA	Georgia	0 - 1
MS	Mississippi	8 - 9
TN	Tennessee	7 - 8

States (ZIP code starts with 4)		Possible Second digit
IN	Indiana	6 - 7
KY	Kentucky	0 - 1 - 2
MI	Michigan	8 - 9
OH	Ohio	3 - 4 - 5

States (ZIP code starts with 5)		Possible Second digit
IA	Iowa	0 - 1 - 2
MN	Minnesota	5 - 6
MT	Montana	9
ND	North Dakota	8
SD	South Dakota	7
WI	Wisconsin	3 - 4

States (ZIP code starts with 6)		Possible Second digit
IL	Illinois	0 - 1 - 2
KS	Kansas	6 - 7
MO	Missouri	3 - 4 - 5
NE	Nebraska	8 - 9

States (ZIP code starts with 7)		Possible Second digit
AR	Arkansas	1 - 2
LA	Louisiana	0 - 1
OK	Oklahoma	3 - 4
TX	Texas	5 - 6 - 7 - 8 - 9

States (ZIP code starts with 8)		Possible Second digit
AZ	Arizona	5 - 6
CO	Colorado	0 - 1
ID	Idaho	3
NM	New Mexico	7 - 8
NV	Nevada	8 - 9
UT	Utah	4
WY	Wyoming	2 - 3

States (ZIP code starts with 9)		Possible Second digit
AK	Alaska	9
CA	California	0 - 1 - 2 - 3 - 4 - 5 - 6
HI	Hawaii	6
OR	Oregon	7
WA	Washington	8 – 9

Table 1. A grouping of states with same first ZIP code digit and all possible second digit(s) for each state [22].

This table shows how the second digit for some states can correspond to a single digit (e.g. Montana – 9), two digits (e.g. Indiana – 6,7), three digits (e.g. Missouri – 3,4,5), four digits (e.g. Virginia – 0,2,3,4), five digits (e.g. Texas – 5,6,7,8,9) or seven digits (e.g. California – 0,1,2,3,4,5,6). Since some of these states share similar sets of possible second digit(s) - for example both Kansas and Indiana have the digits 6 and 7 as possible second digits – we were able to eliminate redundancies among metaclasses and compile a list of metaclasses needed for this model, as shown in Table 2.

After analyzing the testing data found on CEDAR CDROM-1 [1] (the standard test database), we found that the metaclass needed to classify whether the second digit in a California ZIP code is 0, 1, 2, 3, 4, 5 or 6 can be eliminated due to insufficient testing data, as shown in Table 3[5].

ZIP	First Digit									
Second Digit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	5	0	7	10	7	1	9	10	5	0
1	1	7	8	0	1	0	0	1	0	0
2	8	0	8	2	2	7	1	11	7	0
3	6	7	0	0	13	7	11	8	6	0
4	0	5	0	1	0	1	0	3	6	0
5	6	0	4	7	1	7	1	0	6	8
6	5	15	1	2	11	0	13	0	1	10
7	5	1	2	10	0	3	0	12	6	11
8	1	0	5	0	12	6	12	0	2	7
9	1	8	7	9	4	2	1	0	6	13

Table 3. We created a frequency matrix of the test dataset found on CEDAR CDROM-1 [1], according to the first and second digits. A value of n represents n ZIP codes found in the test data with the corresponding column and row numbers as the first and second digits of the ZIP code. For example, there were five ZIP codes in the test dataset beginning with the two digits '1 4'.

#	MLP Metaclass
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	0
11	0 - 1
12	1 - 2
13	2 - 3
14	3 - 4
15	5 - 6
16	6 - 7
17	7 - 8
18	8 - 9
19	0 - 1 - 2
20	2 - 3 - 4
21	3 - 4 - 5
22	4 - 5 - 6
23	0 - 2 - 3 - 4
24	0 - 1 - 2 - 3 - 4
25	5 - 6 - 7 - 8 - 9
26	0 - 1 - 2 - 3 - 4 - 5 - 6
27	0 to 9

Table 2. A list of all MLP metaclasses needed for the model, including the not needed metacl ass numbered as 26.

[5] The testing data found on CEDAR CDROM-1 [1] does not provide ZIP codes for California with 0, 1, 2, 3, or 4 as the second digit. Therefore the 26th metaclass listed in Table 4 is not needed and can be replaced with the one that classifies the digits 5 and 6 instead.

Finally, to classify the third, fourth and fifth digits of a ZIP code, we used the last metaclass listed in Table 2 {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. Recall that Morita and colleagues [16] used five MLP neural networks in their model to classify the five different metaclasses of digits found in Brazilian bank check dates. The performance of their model on four digit years was 100%. On two digit days performance was 93.2% and on two digit years the performance was 97.2%. (The discrepancy between performance on two digit years and two digit days was explained by the fact that year always appears at the end of a date on Brazilian bank checks, which led to an improvement in the segmentation and, thus, recognition of two digit years.) Liu and colleagues [4] also tested a MLP network as one of the classifiers in their model to recognize handwritten ZIP codes. Their model integrated segmentation and a single MLP for recognition and did not make use of domain knowledge (e.g. metaclasses of digits). When trained without outliers, their model's performance was 63.99%. However, when trained with outliers the model's performance was 89.22%. (This compared favorably to the best performance of 90.37% obtained using the LQDF classifier in their model.) Due to the success of MLP classifiers in both studies and the desire to extend the work of both studies, we also used MLP classifiers in our model.

Classifier structure. The input to each network used for classification is a feature vector of length 13. The hidden layer in each network contains an empirically determined number of hidden units, each of which is connected to all input and output units. The number of hidden units for all of the networks is five, with the exception of the last two networks (26, 27) shown in Table 2. The network numbered 26 requires seven hidden units while the one numbered 27 requires nine hidden units. The output layer for each of the MLP networks contains a number of output units corresponding to the number of digits in each metaclass. For instance, a MLP network intended to classify whether a given digit is a 4 (metaclass {4}) must have two output nodes in the output layer, one for the digit 4 and one for any other digit. A MLP network intended to classify a particular digit as 6 or 7 (metaclass {6, 7}) must have three nodes in the output layer, one for the digit 6, one for the digit 7, and one for any other digit, for example.

Training data. To train the classifiers we used the same dataset used by Liu et al. [4]. The classifiers were trained on data compiled from the NIST Special Database 19 (SD19) [21]. We used 66,214 digit samples from the segmented hand-printed digits found in SD19. We also generated 16,000 outlier training patterns from the training digit data. This idea of combining outliers with the training data in order to improve subsequent classification has been successfully used by Liu et al. [4] and by Lauer et al. [8]. We generated outlier patterns, using the technique of Liu et al. [3], by merging and splitting training images. A pair of digit images generated four outlier patterns: full-full combination, full-half combination, half-full combination, and half-half combination as shown in Figure 10.

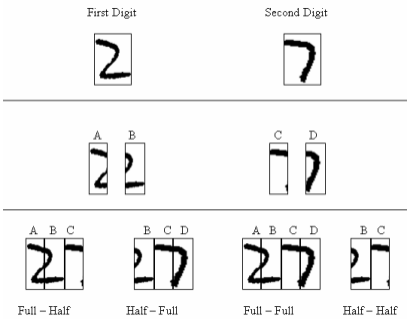

Figure 10. A sample of outliers generated from the NIST SD19 dataset [21].

The patterns were generated by first arbitrarily selecting two digits (see top of

Figure 10). The first digit is split vertically into two parts, A and B. The second digit is also split vertically into two parts, C and D. In order to generate four outliers, parts must be joined together as follows: A, B, C will make one outlier (Full, Half). Then, B, C, D will make another outlier (Half, Full). Then A, B, C, D combined together make another outlier (Full, Full). Finally, B and C combined make still another outlier (Half, Half).

Testing data. The system was tested on 436 5-digit ZIP code images found on CEDAR CDROM-1 in the Binary ZIP code directory (BINZIP). These images were used by Liu et al. [4] to test their classification methods. The images were extracted from live mail images of USPS [1].

Results and Discussion

Our aim was to achieve the following goals: (1) determine if the technique of metaclasses introduced by Morita et al. [16] can be extended to the domain of U.S. ZIP codes, and (2) test the conjecture of Liu et al. [4] that an integrated segmentation and recognition stage is necessary for machine recognition of handwritten ZIP codes. Regarding this latter goal, we tried to keep our model as close as possible to theirs, but without using an integrated segmentation and recognition stage. Instead, we used distinct segmentation and recognition stages. We also incorporated the metaclass technique of Morita et al. [16] (domain knowledge) into our recognition stage. Other than these key differences, we tried to make our model as similar to the model of Liu et al. [4] as possible. For example, our segmentation is similar to their pre-segmentation. Both of our models used MLP classifiers, and our MLP classifiers and their MLP classifier were trained using the same learning method. Furthermore, we used the same base training dataset and included outliers in our training data that were generated by the same method that they used to generate outliers for their training data. Finally both our model and their model were tested on the same test dataset.

The performance of our model was evaluated using the 'test' dataset found on CEDAR CDROM-1 [1]. Since our model assumes prior knowledge of destination state, this information was identified beforehand. To achieve recognition of a given ZIP code, the segmented vectors for a ZIP code, along with the destination state, are passed to the recognition module. This recognition module invokes the appropriate metaclass (MLP network) based on the given state and digit position.

To test our model, we used two different versions. The first version of our model used domain knowledge as has been described. That is, the segmentation stage assumed five digit ZIP codes, and the recognition stage used the digit metaclasses technique. The second version of our model applied domain knowledge in the segmentation stage only (i.e. the knowledge that ZIP codes contain five digits), and it did not assume knowledge of the destination state or ZIP code structure in the recognition stage. That is, it used only a single MLP network which classifies each digit as one of the ten possible digits. The two versions of our model allowed measuring the effectiveness of applying domain knowledge to the recognition of handwritten ZIP codes. Specifically, it allowed us to test whether the metaclasses technique employed by Morita et al. [16] would also succeed in machine recognition of U.S ZIP codes, because the first version of our model used this metaclasses technique, while the second version of our model did not.

The correct recognition rate achieved by the first version using the metaclasses technique was 88.76% with no rejection. The performance of the second version (which did not use the metaclasses technique) was 75.69% with no rejection (see Table 4). Comparing the results of the first version to the second version, it is clear that using domain knowledge, in the

form of the metaclasses technique, improves the recognition of handwritten ZIP codes.

Model	Performance
Version 1 of our model using metaclasses technique (domain knowledge)	88.76%
Version 2 of our model not using metaclasses technique (no domain knowledge)	75.69%
Liu and colleagues MLP model	89.22%

Table 4. Results of our model and that of Liu et al. [4].

Further, these results clearly show that the metaclasses technique used by Morita et al. [16] in machine recognition of Brazilian bank check dates is also quite effective in machine recognition of U.S ZIP codes. Testing two versions of our model – one with domain knowledge and one without – also allowed us to examine the claim by Liu et al. [4] that machine recognition of ZIP codes requires an integrated segmentation and recognition stage. The version of our model which did not use domain knowledge (version 2) was similar to their model with the major exception that their complex integrated segmentation and recognition module was replaced in our model by a recognition module that used just one MLP to classify each of the five digits found by our segmentation module. The performance of the first version of our model with domain knowledge used in the recognition module was 88.76%. Meanwhile the performance of the second version of our model with no domain knowledge in the recognition model was 75.69%. These results show the effectiveness of applying domain knowledge to the recognition of handwritten ZIP codes. Compared to the results of our models, the recognition performance reported by Liu and colleagues [4] was 89.22%, (see Table 4). This implies that when domain knowledge is not used an integrated segmentation and recognition module may be necessary for ZIP code recognition. However, when domain knowledge is used, an integrated segmentation and recognition module may not be necessary.

CONCLUSION

Results of this study have two important implications. First, a complex problem such as recognizing ZIP codes can have a simple, straightforward solution. Recall that Liu and colleagues [4] claimed that such problems could not be solved without having an integrated segmentation and recognition module. We have shown that this is not necessarily true. Our model uses a distinct segmentation module followed by a recognition module employing domain knowledge, and it performed comparably to their model. Compared to their model, ours took a simpler approach to solve the problem. Our segmentation module is straightforward, including some of the same elements of their "pre-segmentation" module. In the recognition module, even though we used 26 neural networks in our model, these networks were not complex and were easy to train.

The second implication is that domain knowledge can aid in the recognition of ZIP codes. Specifically, the effectiveness of the metaclasses technique of Morita et al. [16] extends beyond the recognition of Brazilian bank check dates to recognition of U.S. ZIP codes. Perhaps this technique could also be applied to solve other digit string problems such as dollar amount on checks, and social security number or driver license number on forms. An indication for the effectiveness of domain knowledge in recognizing handwritten ZIP codes was shown by comparing the one version of our model, which used the metaclasses technique in the recognition stage, to a

second version, which did not use the metaclasses technique in the recognition stage. The version which used domain knowledge in the recognition stage had much better recognition performance than the version which did not use domain knowledge. To tie these two discoveries together, the claim of Liu et al. [4] about the need for integrated segmentation and recognition seems reasonable when domain knowledge is not used. However, when domain knowledge is applied, there seems to be no need for a complex model that integrates segmentation with recognition.

REFERENCES

[1] CEDAR CDROM-1 Specifications of the Databases. Retrieved September 8, 2005, from http://www.cedar.buffalo.edu/Databases/CDROM1

[2] C. Gonzalez, E. Woods, Digital Image Processing, second ed., Prentice Hall, New Jersey, 2002.

[3] C.L. Liu, S. Hiroshi, F. Hiromichi, Performance evaluation of pattern classifiers for handwritten character recognition, International Journal on Document Analysis and Recognition (IJDAR) 4, (2002) 191-204.

[4] C.L. Liu, K. Nakashima, H. Sako, H. Fujisawa, Integrated Segmentation and Recognition of Handwritten Numerals: Comparison of Classification Algorithms, International Workshop on Frontiers in Handwritten Recognition (IWFHR), 8 (2002) 303-308.

[5] C.L. Liu, K. Nakashima, H. Sako, H. Fujisawa, Handwritten Digit Recognition Using State-of-the-Art Techniques, Pattern Recognition, 36 (2002) 2271-2285.

[6] C.L. Liu, H. Sako, H. Fujisawa, Effects of Classifier Structures and Training Regimes on Integrated Segmentation and Recognition of Handwritten Numeral Strings, Pattern Analysis and Machine Intelligence, (2004) vol. 26, No. 11. 1395-1407.

[7] D. Bouchaffra, V. Govidaraju, S.N. Sirihari, Recognition of Strings Using Nonstationary Markovian Models: An Application in ZIP Code Recognition. Conference on Computer Vision and Pattern Recognition (CVPR). (1999) 2174-2179.

[8] F. Lauer, C.Y. Suen, G. Bloch, A trainable feature extractor for handwritten digit recognition, Pattern Recognition, 40 (2007) 1816-1824.

[9] H. Rababaah, Asphalt Pavement Crack Classification Using AI and Computer Vision, Masters Thesis, Indiana University, (2005).

[10] I. Chaaban, M.R. Scheessele, Human Performance on the USPS Database. Tech. Rep. IUSB (2007) TR-20070619-1.

[11] I. Chaaban, M.R. Scheessele, Human Performance in Recognition of Handwritten ZIP Codes from the CEDAR Database. Tech. Rep. IUSB (2008) TR-20080805-1

[12] J.G. Etgen, Salas and Hille's Calculus: One Variable, eighth ed., John Wiley & Sons, INC. 1999.

[13] J. Mead, Speed Reading, Journal of University of Buffalo Research, (1991) vol1.1 page 2 and 3. Retrieved April 8, 2005, from http://www.cedar.buffalo.edu/pub_docs/article41.html.

[14] L.G. Shapiro, C.G. Stockman, Computer Vision, second ed., Prentice Hall, New Jersey, 2002.

[15] L.O. Zhang, C.Y. Suen, Recognition of Courtesy Amounts of Bank Checks based on a Segmentation Approach, International Workshop on Frontiers in Handwriting Recognition (IWFHR), (2002) 298-302.

[16] M. Morita, R. Sabourin, F. Bortolozzi, C.Y. Suen, Segmentation and recognition of handwritten dates: a HMM-MLP hybrid approach, International Journal on Document Analysis and Recognition (IJDAR) 6, (2004) 248-262.

[17] P.Y. Simard, D. Steinkraus, J.C. Platt, Best practices for convolutional neural networks applied to visual document analysis, in: Proceedings of the Seventh International Conference on Document Analysis and Recognition, vol. 2, Edinburgh, Scotland, (2003) 958-962.

[18] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, second ed., Prentice Hall, New Jersey, 2002.

[19] T. Kohonen, The Self-Organizing Map, Proc. IEEE (1990) vol. 78, 1464-1480.

[20] X. Fan, G. Fan, D. Liang, Joint Segmentation and Recognition of License Plate Characters, Image Processing. IEEE (2007) vol. 4, IV-353-IV – 356.

[21] Y. LeCun, The MNIST database of handwritten digits. Retrieved March 15, 2006, from http://yann.lecun.com/exdb/mnist/index.html.

[22] ZIP-Codes. Retrieved March 15, 2006, from http://www.zip-codes.com

ACKNOWLEDGEMENTS

A special thanks to Dr. Michael R. Scheessele, Dr. James Wolfer, and Dr. Yi Cheng , for their time and useful comments.