# Chapter 7

# Character Recognition with Invariance Signature Networks

## 7.1 Retention of Sufficient Information

In Chapter 5 a new class of invariant signature for two dimensional contours was derived. It was established that these Invariance Signatures are themselves invariant under rotations, dilations and translations of the contour. In Chapter 6 it was shown that a MBNN can be constructed which computes these signatures and classifies contours on that basis.

It remains now to demonstrate that these Invariance Signatures retain enough information content to be usable for pattern recognition, and that they are not unduly sensitive to noisy data such as that encountered in real applications. In order to do this, the system is applied to the classification of letters of the Roman alphabet, both for "perfect"[1] machine-generated training and test data, and for data gathered using physical sensors.

## 7.2 Perfect Data

### 7.2.1 Departures from Exact Invariance

Despite the fact that Invariance Signatures are provably invariant under rotations, dilations and shifts in the plane when calculated for a continuous contour, departures from invariance occur in real applications in several ways. Scanned data contains random noise from the sensor, though the quality of scanners now available renders this negligible for this application. More important sources of error are discussed below.

[1] Throughout this chapter, the term "perfect" will be used to describe data which is both noise-free and unambiguous.

**Quantization Noise** Noise is introduced into the tangent estimation procedure by the sampling of the contour. Since the estimated tangent orientation is quantized,[2] the value of the Local Measure of Consistency $\iota_G$ can be changed when a contour is quantized at a new orientation. It is possible to compensate partially for this effect by using sufficiently wide bins when calculating the Invariance Signature $I(\iota_G)$, but there will still be errors when the change in estimated orientation moves the resultant $\iota_G$ across bin boundaries.

**Ambiguous Characters** In many fonts some letters are rotated or reflected versions of other letters, such as {b, d, p, q} and {n, u} in many sans serif fonts. In fonts with serifs, the serifs are often so small that they make a negligible contribution to the Invariance Signature. Consequently, it is impossible to classify isolated characters into 26 classes if shift, rotation, scale and reflection invariance is desired. Admittedly, reflection invariance is not usually desired in a character recognition system, but it is an unavoidable characteristic of the ISNNC. In commercial optical character recognition systems, context information is used to resolve ambiguous letters,[3] which occur even in systems without inherent invariances. Such an approach would be equally applicable as a post-processing stage for the ISNNC system.

### 7.2.2 The Data Set

These effects can be avoided by using a computer to produce a "perfect" data set, which is free from quantization noise and contains no letters which are transformed version of others in the data set. Such a data set can be used to demonstrate that Invariance Signatures retain sufficient information for classification in the absence of noise, and these results can be used as a basis for assessing the performance of the system on real data.

A training data set was created using a screen version of the Helvetica font. Only the letters {a, b, c, e, f, g, h, i, j, k, l, m, n, o, r, s, t, v, w, x, y, z} were used, so that ambiguity was avoided. An $18 \times 18$ binary image of each letter was produced. This training data set is shown in Figure 7.1.

A "perfect" test data set was created by computing reflected and rotated versions of the training data, where all rotations were by multiples of $\frac{\pi}{2}$ radians, so that there was no quantization error. This test data set is shown in Figure 7.2.

[2] See §6.1.2.

[3] *i.e.* surrounding letter classifications and a dictionary of acceptable words.

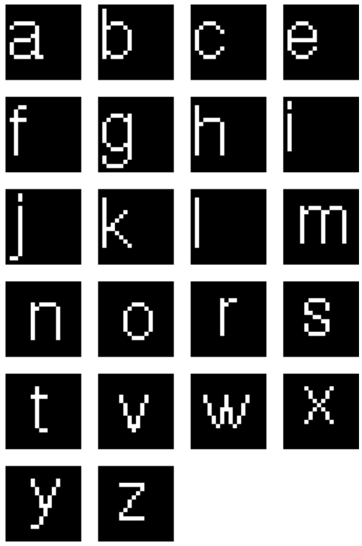



Figure 7.1: Training set of canonical examples of unambiguous characters.

### 7.2.3 Selected Networks Applied to this Problem

Simulations indicated that the training set shown in Figure 7.1 could be learnt by a neural network with no hidden layer: it is a linearly separable problem [Minsky and Papert, 1969]. It was assumed that this would also be true of the Invariance Signatures calculated from these data.[4] Consequently, two different network architectures were constructed for comparison.[5] The first was a TNN with a $18 \times 18$ input layer, no hidden layers, and a $1 \times 22$ output layer. The other was an ISNNC, with an $18 \times 18$ input layer for the Invariance Signature calculation stage. The Invariance Signature was calculated using 5 bins for each Lie transformation, meaning that the Invariance Signature layer had $3 \times 5$ nodes. This was connected directly to a $1 \times 22$ output layer, with no intervening hidden layers, forming an linear classifier sub-network.

### 7.2.4 Reduction of Data Dimensionality

It should be noted that although the Invariance Signature Calculation stage of the ISNNC has to be run every time a new pattern is to be classified at run time, it is only necessary to calculate the Invariance Signatures of the training and test data once. The final classification stage of the ISNNC can then be trained as a separate module. This can lead to an enormous reduction in the time taken to train a network, since the number of training passes required to learn a large and complex data set is typically

[4]This was confirmed by subsequent experiments.

[5]All networks described were constructed, trained and evaluated using the **Xnet** simulator described in Appendix B.

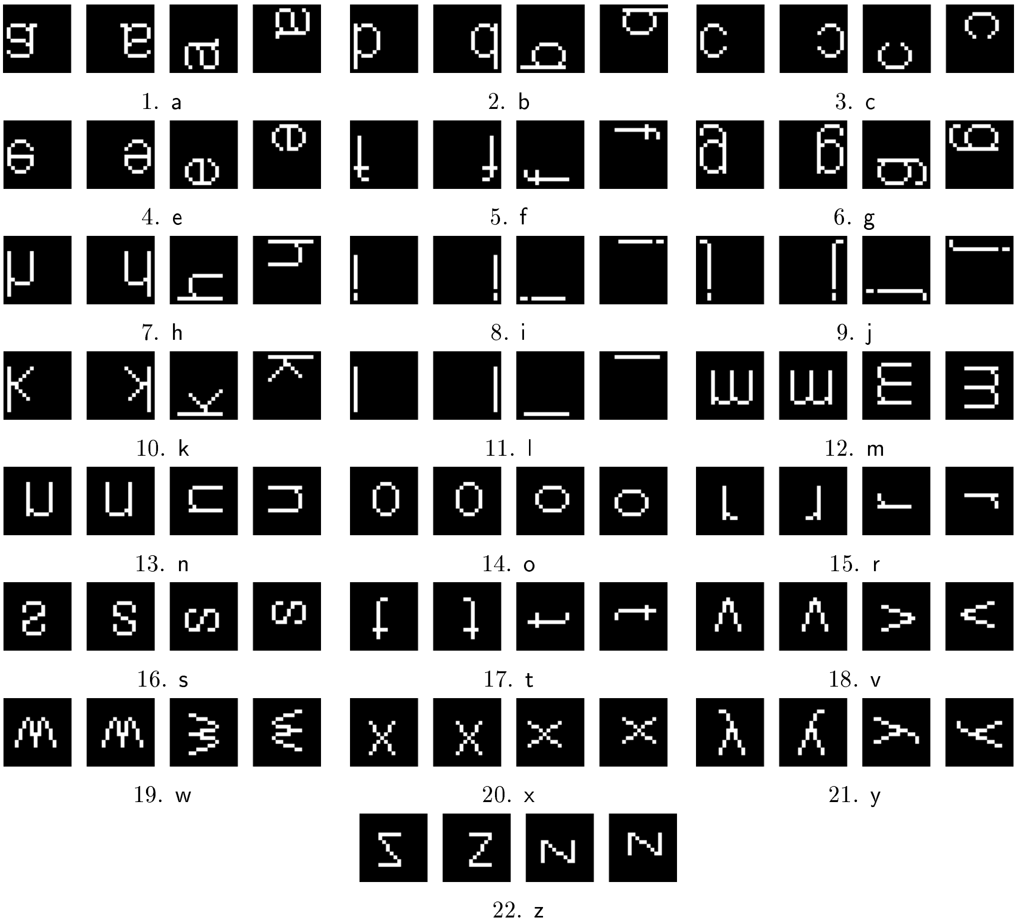



Figure 7.2: Test set of ideally shifted, rotated and reflected letters.

order $10^2$ to $10^3$, and the total number of parameters $n_p$ in a TNN is:

$$n_p = \sum_{i=1}^{N-1} (\text{nodes in layer})_{i-1} \times (\text{nodes in layer})_i \tag{7.1}$$

where $N$ is the total number of layers, and $i$ is the layer number, where the input layer is labeled 0. The iteration time during training is proportional to $n_p$, so, for this example, each training iteration for the Invariance Signature classification stage will be $\frac{18\times18}{3\times5} = 21.6$ times faster than that for the TNN.

The calculation of the Invariance Signatures is admittedly time-consuming, but the time is made up many times over during training when the input image size is large. In real applications, the image size is typically larger than the $18 \times 18$ image used in this demonstration. ISNNC systems can therefore provide an extremely significant reduction in the dimensionality of the training data, and a corresponding reduction in the time taken to train the classification network. Moreover, the simulation of the ISNNC on a sequential computer is unable to take advantage of the inherently parallel, local computations that characterize many of the ISNNC modules. A parallel implementation would largely alleviate the time-consuming nature of the Invariance Signature calculation.

### 7.2.5 Perfect and Network-Estimated Local Orientation

Since the neural network implementation of the Local Orientation Unit described in §6.1.2 still had some residual error, two versions of the Invariance Signature training and test data were created, one with the local orientation at each point calculated directly from the covariance matrix eigenvectors, and the other using the neural network module. Results from these data sets will be compared to evaluate the importance of accurate orientation extraction in the neural module.

Ten examples of each network were made, each with a different initialization of parameters, so that it could be ensured that the results were reproducible. The TNNs and the ISNNC linear classifier modules were all trained using the backpropagation algorithm, with the weight update equation shown in Equation 4.14. The parameters were $\epsilon = 0.5$ and $\alpha = 0.5$ for the Invariance Signature classification modules, and $\epsilon = 0.025$ and $\alpha = 0.5$ for the TNNs. The difference in $\epsilon$ is to compensate for the different fan-in to the output nodes in the two networks. Training was continued for 1000 iterations. A network was deemed to have classified a pattern as belonging to the class corresponding to the output node which had the highest output value, so no patterns were rejected.

#### Results for Traditional Neural Networks

The results obtained with TNNs are summarized in Table 7.1. It is clear that the TNNs do not exhibit transformation invariance. They could not be expected to do so, since no transformed versions of the patterns were included in the training data.

It might be expected that the fact that the final performance of the TNNs is better than chance ($\frac{1}{22} = 4.5454\%$) is because some of the transformations of the training data resulted in patterns almost identical to the untransformed training pattern for some of the highly symmetrical letters in the training set (*e.g.*o, s, x and z). Analysis of which test patterns were classified correctly, however, shows that this is not the case. The 12 correctly classified test patterns are shown in Figure 7.3. No reason for these particular patterns being classified correctly is obvious. It is well-known that the "rules" generated by TNNs are hard to extract, although progress has been made in that area [*e.g.* Wiles and Ollila, 1993; Wiles and Elman, 1995]. What is clear, however, is that chance must play a part in some of these classifications, for instance i4, which shares no "on" pixels with the training example of i (see Figures 7.1 and 7.3).

Table 7.2 shows that marginally better performance on the test data could have

| | Best Performance | | Final Performance | |
|---|---|---|---|---|
| | Training Data | Test Data | Training Data | Test Data |
| Network 1 | 100.00 | 14.773 | 100.00 | 13.636 |
| Network 2 | 100.00 | 14.773 | 100.00 | 13.636 |
| Network 3 | 100.00 | 15.909 | 100.00 | 13.636 |
| Network 4 | 100.00 | 14.773 | 100.00 | 13.636 |
| Network 5 | 100.00 | 14.773 | 100.00 | 13.636 |
| Network 6 | 100.00 | 15.909 | 100.00 | 13.636 |
| Network 7 | 100.00 | 14.773 | 100.00 | 13.636 |
| Network 8 | 100.00 | 14.773 | 100.00 | 13.636 |
| Network 9 | 100.00 | 14.773 | 100.00 | 13.636 |
| Network 10 | 100.00 | 14.773 | 100.00 | 13.636 |
| $\mu \pm \sigma$ | $100.00 \pm 0.000$ | $15.000 \pm 0.454$ | $100.00 \pm 0.000$ | $13.636 \pm 0.000$ |

Table 7.1: Classification performance (% correct) of traditional neural network classifiers trained for 1000 iterations with the data shown in Figure 7.1 and tested with the "perfect" data set in Figure 7.2.

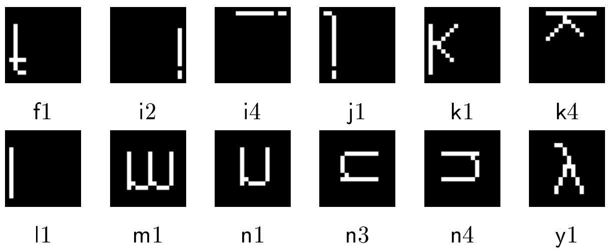



Figure 7.3: Test patterns classified correctly by the TNNs.

been achieved by employing an early-stopping scheme in which training was stopped when classification performance on the test set started to deteriorate. This would, however, have been at the cost of less than 100% correct performance on the training data. It should be noted that the error on the test set, as defined in Equation 1.3, decreased throughout the entire 1000 iterations. This indicates that an early-stopping scheme based on the test set error would fail to improve classification performance in this case. Figure 7.4 shows how the classification performances and errors typically varied during training.

It should be acknowledged that the TNNs could not really be expected to perform better than chance on this test set given the training set used. Their architecture provides no invariances, and generalization cannot be expected unless multiple transformed versions of the patterns are included in the training set. This argument can be used against all the comparisons between TNNs and MBNNs in this thesis: they are not really fair. Nevertheless, these comparisons between naive applications of TNNs and specifically-designed MBNNs demonstrate that MBNNs can perform successfully using training sets completely inadequate for TNNs. Moreover, these MBNNs are of

| | Iteration | % Correctly Classified | |
|---|---|---|---|
| | | Training Data | Test Data |
| Network 1 | 86 | 100.00 | 14.773 |
| Network 2 | 33 | 95.455 | 14.773 |
| Network 3 | 1 | 90.909 | 15.909 |
| Network 4 | 85 | 100.00 | 14.773 |
| Network 5 | 94 | 100.00 | 14.773 |
| Network 6 | 1 | 90.909 | 15.909 |
| Network 7 | 95 | 100.00 | 14.773 |
| Network 8 | 1 | 95.455 | 14.773 |
| Network 9 | 86 | 100.00 | 14.773 |
| Network 10 | 19 | 95.455 | 14.773 |
| $\mu \pm \sigma$ | $50 \pm 40$ | $96.818 \pm 3.550$ | $15.000 \pm 0.454$ |

Table 7.2: Performance at iteration at which best performance on test data occurred for traditional neural network classifiers trained with the data shown in Figure 7.1 and tested with the "perfect" data set in Figure 7.2.

lower dimensionality than the TNNs. Providing with the TNNs with sufficiently large training sets would only make their training still more computationally-expensive, with no *guarantee* of invariant performance.

### Results with Perfect Local Orientation

The results for the ten ISNNCs which used perfect Local Orientation Extraction are shown in Table 7.3. The average number of iterations for 100% correct classification to be achieved was 220. Since the problem is linearly-separable, it could in fact be solved directly, using a linear equations technique such as singular-valued decomposition [Press et al., 1992]. Since weights may set by any method at all in the MBNN paradigm, this makes the comparison of convergence times somewhat irrelevant. Nevertheless, the MBNN modules, although taking, on average, 4.4 times as many iterations to converge as the TNNs, were still 4.9 times faster to train, due to their lower dimensionality, as discussed in §7.2.4.

As expected, the ISNNCs generalize perfectly to the transformed images in the test set. The network architecture constrains the system to be shift- rotation-, scale- and reflection-invariant in the absence of quantization noise, so this is no surprise. Importantly, the result indicates that sufficient information is retained in the 5 bin Invariance Signatures for all 22 unambiguous letters of the alphabet to be distinguished. Inspection of the percentage sum squared error values after each iteration indicated that the error on the test set was indeed identical to that on the training set: for perfect data, the ISNNC produces perfect results.

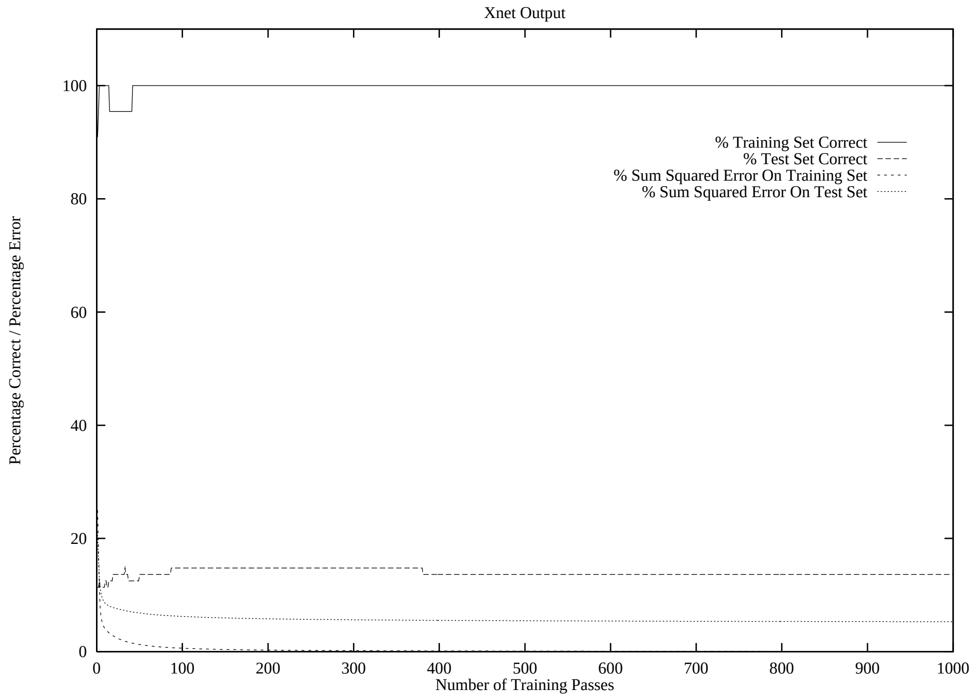



Figure 7.4: Classification performance and errors of TNN 2 during training.

### Results with Network-Estimated Local Orientation

The perfect results above were obtained using a hybrid system, which used a module which was not a neural network to calculated the local orientation at each point. The neural module for local orientation extraction used here was trained using backpropagation until 98.93% of the variance in the training data was explained (see Equation 6.7). The Invariance Signatures for the test and training data were calculated using the system incorporating this module, and the classification module was then trained separately using these Invariance Signatures. Systems were produced with both 5 and 10 bin Invariance Signatures. The results obtained are shown in Tables 7.4 and 7.5.

The misclassified patterns for the 5 bin ISNNCs were those shown in Figure 7.5. `t1` and `t2` were classified as `f`, which is understandable, since there is very little difference indeed between the patterns. `t4` was misclassified as `a`. All ten networks had these same misclassifications.

These data show that small residual error in the neural Local Orientation Extraction module does cause a degradation in the transformation invariant classification performance of the ISNNCs. These results are, however, still far superior to those for the TNNs. Moreover, there is no reason that the Local Orientation Extraction module could not be trained further. It is to be expected that the invariant classification performance would continue to approach 100% as the accuracy of the module was improved.

| | Best Performance | | Final Performance | |
|---|---|---|---|---|
| | Training Data | Test Data | Training Data | Test Data |
| Network 1 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 2 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 3 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 4 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 5 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 6 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 7 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 8 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 9 | 100.00 | 100.00 | 100.00 | 100.00 |
| Network 10 | 100.00 | 100.00 | 100.00 | 100.00 |
| $\mu \pm \sigma$ | 100.00 $\pm$ 0.00 | 100.00 $\pm$ 0.00 | 100.00 $\pm$ 0.00 | 100.00 $\pm$ 0.00 |

Table 7.3: Classification performance (% correct) of Invariance Signature Neural Network Classifiers (with perfect Local Orientation Extraction) trained for 1000 iterations with the data shown in Figure 7.1 and tested with the "perfect" data set in Figure 7.2.

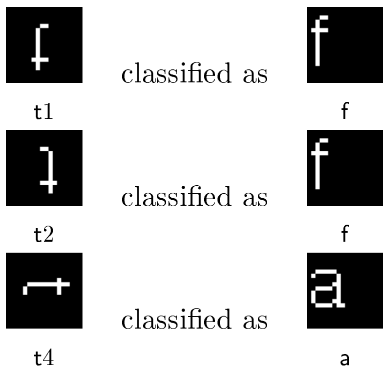



Figure 7.5: Test Patterns Misclassified by the 5 Bin Invariance Signature Neural Network Classifiers, and the training examples as which they were incorrectly classified.

The results for the 10 bin system in Table 7.5 show that the effects of inaccuracies in the Local Orientation Extraction module are greater when the number of bins is increased. This is due to the fact that the errors can cause the consistency measure at a point to change bins more easily this way, thus altering the histogram Invariance Signature.

## 7.3 Optical Character Recognition

Having demonstrated that Invariance Signatures retain sufficient information for the classification of "perfect" character data, it is now necessary to show that the system can be used to achieve transformation invariant recognition of data in the presence of sensor and quantization noise. To this end, it was decided to apply ISNNCs to the classification of scanned images of shifted and scaled printed alphabetic characters.

| | Best Performance | | Final Performance | |
|---|---|---|---|---|
| | Training Data | Test Data | Training Data | Test Data |
| Network 1 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 2 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 3 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 4 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 5 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 6 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 7 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 8 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 9 | 100.00 | 96.591 | 100.00 | 96.591 |
| Network 10 | 100.00 | 96.591 | 100.00 | 96.591 |
| $\mu \pm \sigma$ | $100.00 \pm 0.00$ | $96.591 \pm 0.00$ | $100.00 \pm 0.00$ | $96.591 \pm 0.00$ |

Table 7.4: Classification performance (% correct) of 5 Bin Invariance Signature Neural Network Classifiers (with neural Local Orientation Extraction) trained for 1000 iterations with the data shown in Figure 7.1 and tested with the "perfect" data set in Figure 7.2.

### 7.3.1 The Data Set

The data set used was derived from the set of shifted and rotated versions of the lowercase Roman alphabet shown in Figure 7.6. Each character appears in 18 different orientations, at rotation increments of 20 degrees. The shifts arise due to the fact that characters are extracted using a bounding-box technique which takes no account of the centroid position.

An A4 page with these characters printed onto to it from a laser printer at 300 dots per inch was scanned at 75 dots per inch using a UMAX Vista-S6 Scanner. The connected regions in this image were then detected, and the minimum bounding-box which could contain the largest of the characters was calculated ($56 \times 57$ pixels). The image was then segmented into separate characters, and each character was thinned using an algorithm due to Chen and Hsu (1988). The thinning was necessary since the Invariance Signature method is only applicable to thin contours. This set of 468 characters was partitioned into a training set and a test set. The training set consisted of the characters rotated by angles in the range $[0°, 160°]$ relative to the upright characters, and the test set of those in the range $[180°, 340°]$. In Figure 7.6, those characters to the left of the dashed line were used for training, and those to the right for testing. The resultant scanned, extracted and thinned training set is shown in Figures 7.7 and 7.8, and the test set in Figures 7.9 and 7.10, respectively.

It is interesting to compare the subjective visual similarity between the Invariance Signatures both within and between classes for these extracted data. Figure 7.11 shows the first four training examples of the letter "a", accompanied by images showing the tangent estimates at each point of the contours. The tangent estimate is represented

| | Best Performance | | Final Performance | |
|---|---|---|---|---|
| | Training Data | Test Data | Training Data | Test Data |
| Network 1 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 2 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 3 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 4 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 5 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 6 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 7 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 8 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 9 | 100.00 | 95.455 | 100.00 | 93.182 |
| Network 10 | 100.00 | 95.455 | 100.00 | 93.182 |
| $\mu \pm \sigma$ | 100.00 ± 0.00 | 96.591 ± 0.00 | 100.00 ± 0.00 | 96.591 ± 0.00 |

Table 7.5: Classification performance (% correct) of 10 Bin Invariance Signature Neural Network Classifiers (with neural Local Orientation Extraction) trained for 1000 iterations with the data shown in Figure 7.1 and tested with the "perfect" data set in Figure 7.2.

by a line segment with the orientation of the estimated tangent. These images do not indicate the weight assigned to each estimated tangent.

It is not easy to interpret the similarity between these tangent representations of the contours. For this, it is necessary to see the Invariance Signatures. Figure 7.12 shows the Invariance Signatures for the patterns in Figure 7.11 with respect to rotation, dilation and translation. The Invariance Signatures are represented by the Invariance Measure Density histograms. It is immediately apparent that there is a great deal of (subjective) similarity between these representations of the shifted and rotated patterns. They are not identical, as a result of the various sources of noise discussed in §7.2. For classification purposes, however, all that is required is that these signatures be more similar to each other within a letter class than between letter classes. This must be determined by experiment.

Figures 7.13 and 7.14 show the equivalent data for the first four training patterns for the letter "x". These again show the marked within-class similarity, and are also distinctly different to the signatures for the letter "a": these letters were deliberately chosen since "a" is "quite rotationally-invariant", whereas "x" is "quite dilationally-invariant".

### 7.3.2 Selected Networks Employed for this Problem

The methodology employed for this experiment was identical to that used for the synthetic data, described in §7.2.3. The TNN used in this case had a $56 \times 57$ node input layer, and a $1 \times 26$ output layer, giving a massive 83018 independent parameters to be estimated. With only 234 training patterns and 83018 parameters, this problem is almost certain to be linearly separable. Simulation verified that this problem was indeed learnable by a network with no hidden layer.

A variety of different classification modules was tried, for ISNNCs with both 5 and 10 bin Invariance Signatures. These included the simple linear classifier, and a variety of MLP classifiers with differing numbers of nodes in their hidden layers. These experiments indicated that 5 bin Invariance Signatures were insufficient for this problem, and that it was not linearly separable. It also became clear that the errors introduced by the slightly inaccurate network-estimated Local Orientation Extraction caused a noticeable departure from invariance (see §7.2.5). For these reasons, the results presented are for 10 bin ISNNCs with directly-calculated ("perfect") Local Orientation and a MLP classification module. The MLP classifier had a $3 \times 10$ node input layer, a 15 node hidden layer, and a 26 node output layer. This classifier has only 881 parameters to be estimated, a reduction of 99% compared to the TNN linear classifier. This translates to a dramatic reduction in both the storage space and the training time required. Only five TNNs were trained, partly because of the training time needed, and partly because the results were so consistent.

### 7.3.3 Results for Traditional Neural Networks

It might have been expected that the TNNs would perform better on this task than on the task with the synthetic data described in §7.2.3, since this training set does indeed contain differently transformed versions of the canonical untransformed characters. As can be seen from Table 7.6, the results are in fact similar, and slightly worse in this case (average best percent correct of $13.932 \pm 0.300$ compared with $15.000 \pm 0.454$). Although better than chance (3.85% correct), these results are no where near usable for an optical character recognition system.

| | Best Performance | | Final Performance | |
|---|---|---|---|---|
| | Training Data | Test Data | Training Data | Test Data |
| Network 1 | 100.00 | 13.675 | 100.00 | 11.111 |
| Network 2 | 100.00 | 14.103 | 100.00 | 11.111 |
| Network 3 | 100.00 | 14.103 | 100.00 | 10.684 |
| Network 4 | 100.00 | 14.103 | 100.00 | 10.684 |
| Network 5 | 100.00 | 13.675 | 100.00 | 11.111 |
| $\mu \pm \sigma$ | $100.00 \pm 0.00$ | $13.932 \pm 0.300$ | $100.00 \pm 0.00$ | $10.940 \pm 0.209$ |

Table 7.6: Classification performance (% correct) of Traditional Neural Network Classifiers trained for 200 iterations with the data shown in Figures 7.7 and 7.8 and tested with data set in Figures 7.9 and 7.10.

### 7.3.4 Results for Invariance Signature Neural Network Classifiers

The results obtained with ISNNCs attempting to classify the characters into 26 classes appear in Table 7.7. These results show that the ISNNCs achieve a much higher correct classification rate on the test set than the TNNs. The failure of the ISNNCs to achieve 100% correct classification of the training set is in fact not surprising. The test and training sets used for this problem have each individual character of the alphabet mapped to a separate class. Yet, as was discussed in §7.2, the sets of characters {b, d, p, q} and {n, u} are identical under rotations and reflections: transformations under which the ISNNC output is invariant. The expected training set performance for noise-free data is thus $100 \times (20 + 0.25 \times 4 + 0.5 \times 2)/26 = 85\%$ correct. Any performance on the training data better than this must be the result of the fitting of noise in the training data.

| | Best Performance | | Final Performance | |
|---|---|---|---|---|
| | Training Data | Test Data | Training Data | Test Data |
| Network 1 | 95.726 | 72.650 | 95.726 | 71.795 |
| Network 2 | 96.154 | 72.222 | 96.154 | 70.940 |
| Network 3 | 96.154 | 73.077 | 96.154 | 69.658 |
| Network 4 | 95.299 | 73.932 | 95.299 | 70.513 |
| Network 5 | 94.872 | 71.795 | 94.872 | 71.368 |
| Network 6 | 95.726 | 72.650 | 95.726 | 68.803 |
| Network 7 | 96.154 | 73.504 | 96.154 | 69.658 |
| Network 8 | 97.436 | 72.222 | 97.436 | 70.513 |
| Network 9 | 94.444 | 74.359 | 94.444 | 69.658 |
| Network 10 | 96.154 | 70.940 | 96.154 | 70.940 |
| $\mu \pm \sigma$ | 95.812 ± 0.783 | 72.735 ± 0.971 | 95.812 ± 0.783 | 70.385 ± 0.877 |

Table 7.7: Classification performance (% correct) of 10 bin Invariance Signature Neural Network Classifiers (with perfect Local Orientation Extraction) trained for 40000 iterations with the data shown in Figures 7.7 and 7.8 and tested with data set in Figures 7.9 and 7.10.

In order to assess the magnitude of this effect, training and test sets were created in which {b, d, p, q} were assigned the same label, as were {n, u}. The results are shown in Table 7.8. These results show that the average final test set performance was improved by 14.8% by this re-labeling, which is very close to the maximum possible 15.4% achievable if this were the only source of error.

The residual difference between training and test set error is generalization error, rather than invariance error. These networks were trained with only 9 examples of each character, and these examples are quite noisy. There is the unavoidable quantization noise, but there are also some quite marked artifacts, such as the loops introduced by the thinning algorithm, one of which can be seen in pattern a03 in Figure 7.12, and in several patterns in both the training and test sets in Figures 7.7 though 7.10. There

| | Best Performance | | Final Performance | |
|---|---|---|---|---|
| | Training Data | Test Data | Training Data | Test Data |
| Network 1 | 98.718 | 87.179 | 98.718 | 87.179 |
| Network 2 | 99.145 | 83.761 | 99.145 | 82.906 |
| Network 3 | 99.573 | 86.752 | 99.145 | 85.897 |
| Network 4 | 99.573 | 86.752 | 99.145 | 85.043 |
| Network 5 | 98.718 | 88.034 | 98.718 | 88.034 |
| Network 6 | 99.573 | 85.897 | 99.145 | 85.470 |
| Network 7 | 100.00 | 85.897 | 100.00 | 83.761 |
| Network 8 | 98.291 | 86.325 | 98.291 | 85.043 |
| Network 9 | 97.863 | 86.752 | 97.863 | 85.043 |
| Network 10 | 99.145 | 84.615 | 99.145 | 83.761 |
| $\mu \pm \sigma$ | $99.056 \pm 0.628$ | $86.196 \pm 1.179$ | $99.056 \pm 0.628$ | $85.214 \pm 1.483$ |

Table 7.8: Classification performance (% correct) of 10 bin Invariance Signature Neural Network Classifiers (with perfect Local Orientation Extraction) trained for 10000 iterations with the data shown in Figures 7.7 and 7.8 and tested with data set in Figures 7.9 and 7.10, modified to label characters which can be transformed into each other as the same character.

are also two erroneous test patterns: the result of clipping in the segmentation process. They are the final m and the final w. These were left in, as such errors can and do occur in practical applications.

| Misclassifications | | | | | | |
|---|---|---|---|---|---|---|
| d → n | f → t | f → j | i → l | i → l | i → l | i → l |
| i → l | i → l | j → r | k → f | k → r | k → f | l → i |
| m → h | n → b | n → b | q → n | r → f | r → y | r → k |
| r → i | t → f | t → f | t → f | t → f | u → h | w → m |

Table 7.9: Failure analysis for Network 5 from Table 7.8.

The errors made by the ISNNCs are not random. To illustrate this, a failure analysis is presented for Network 5 from Table 7.8, showing how the test patterns were misclassified. Patterns which are perceptually similar are responsible for many of the misclassifications. This means that prior information about likely errors could be used in conjunction with these classifications to aid error correction. This failure analysis indicates that the ISNNCs often appear to make errors that are very "human", which is a promising indication that the Invariance Signature measures contour similarity in a way similar to the (unknown) measure used by humans. Inspection of the actual thinned patterns used for training and testing the networks indicates that the patterns for the letters {i, j, l} are little more than straight lines. If these were to be re-labeled as the same character, final test set performance for Network 5 would improve to 91.026% correct. If the same were done for {f, t}, which are also extremely similar, test set performance would be 93.162% correct.

The misclassifications of patterns to the classes b and n may be due to the fact that the re-labeled dataset implies a non-uniform prior probability distribution for these classes: b was used as the target class for input patterns corresponding to {b, d, p, q}, and thus occurs four times more frequently in the training data than any other class. Similarly, n was used as the target for inputs {n, u}. As discussed in §3.2.6, networks will tend to "guess" these classes more frequently than the others.

In the context of the extremely small training set, this is a remarkable result, comparable with character recognition results achieved by others with thousands, rather than tens, of training patterns. Such re-relabeling is in any case an essential feature of a *truly* invariant optical character recognition system, since some characters are inherently ambiguous under rotation and reflection. Others are extremely similar, and noise can render them indistinguishable. In any real optical character recognition system, a dictionary is used to verify recognized words, and context is used to correct incorrectly labeled patterns. Another possibility is that the orientation of correctly-classified unambiguous characters could be used to infer the correct labeling of ambiguous characters.

Perhaps most importantly, these results show that ISNNCs can classify correctly patterns which have been transformed by arbitrarily large amounts. The shifts and rotations of the input images are not restricted to small perturbations of the example patterns. This indicates that the ISNNCs are performing truly invariant pattern recognition, rather than interpolation-based generalization.

This study should be considered to be a "proof of concept", both for the Invariance Signature as a contour descriptor, and for MBNNs which classify on that basis. It is not intended to be a large-scale experiment which corresponds to a real application. Such a study would require much greater quantities of data than are used here. We believe, however, that the experiments presented here are sufficient to demonstrate that the theoretical work of Chapters 5 and 6 has resulted in a system that is genuinely useful for robust invariant pattern recognition. The quality of the results obtained is in fact remarkable when the size of the training sets is compared to those used in other neural network approaches to the optical character recognition problem, which frequently consist of many thousands of input-output examples.

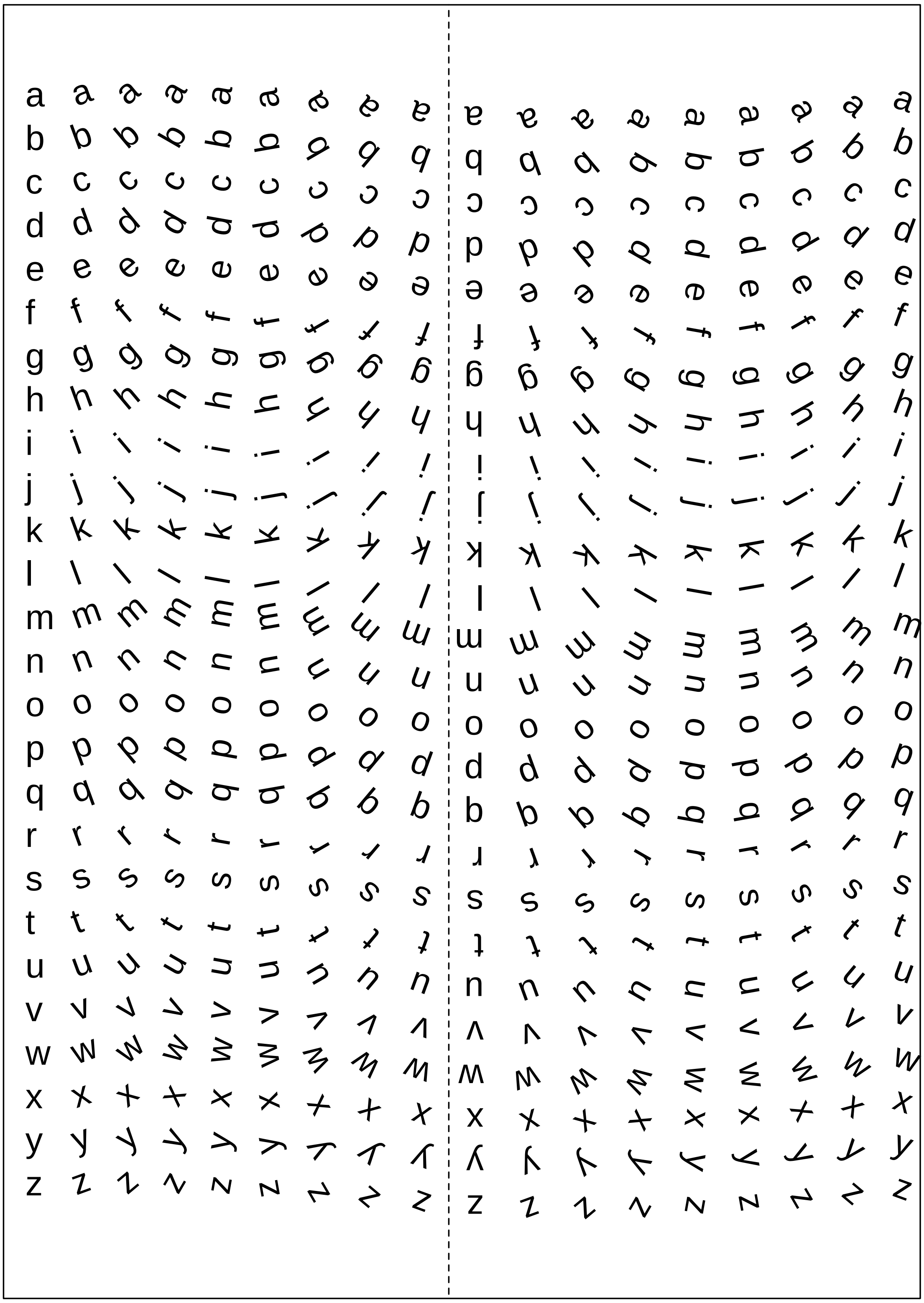


Figure 7.6: The characters scanned to generate the real data set. The box shows the border of an A4 page (210mm × 297mm) so that the size of the original characters may be judged. The dashed line shown the partition into training and test data.

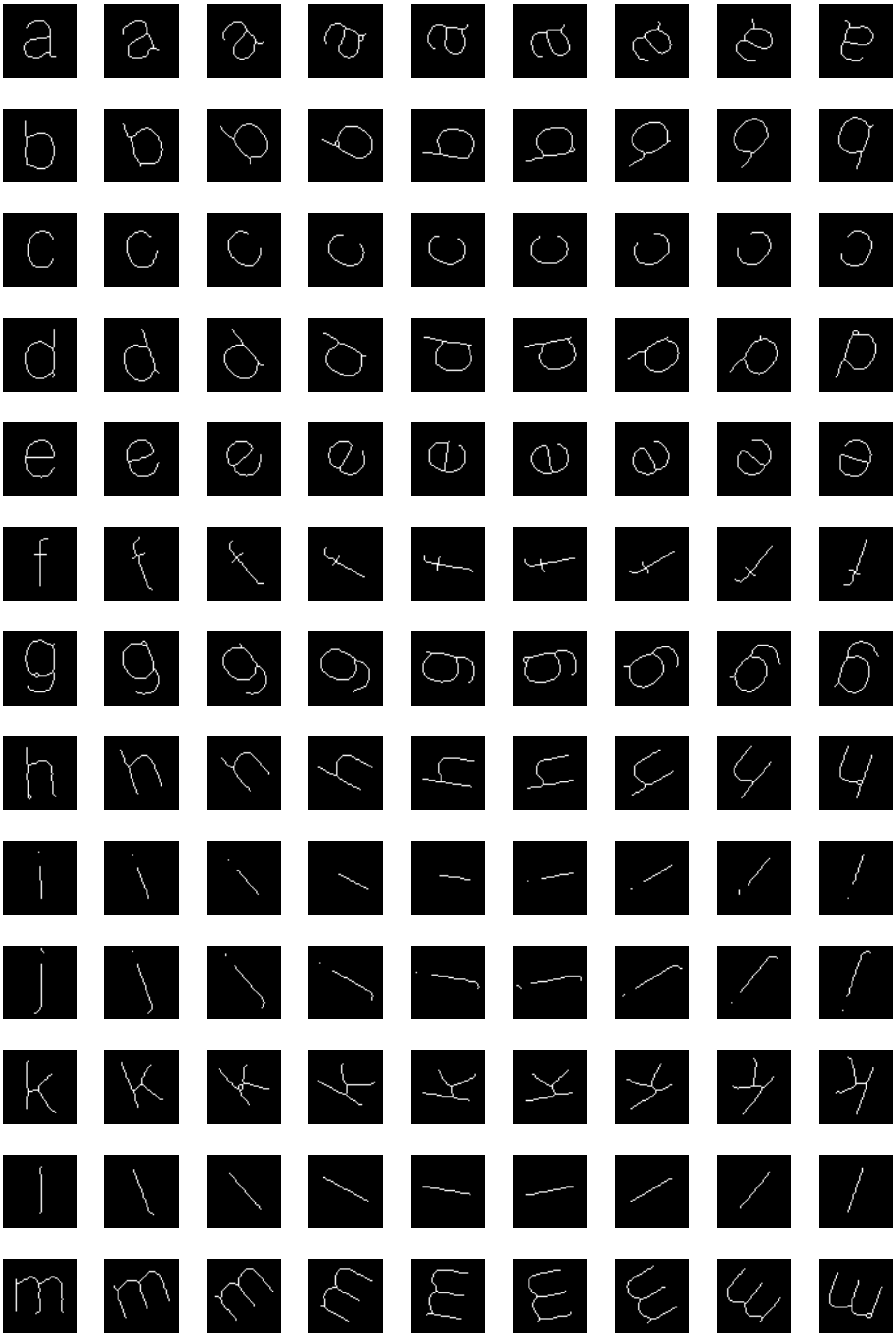


Figure 7.7: Training set of thinned, scanned characters (Part 1).

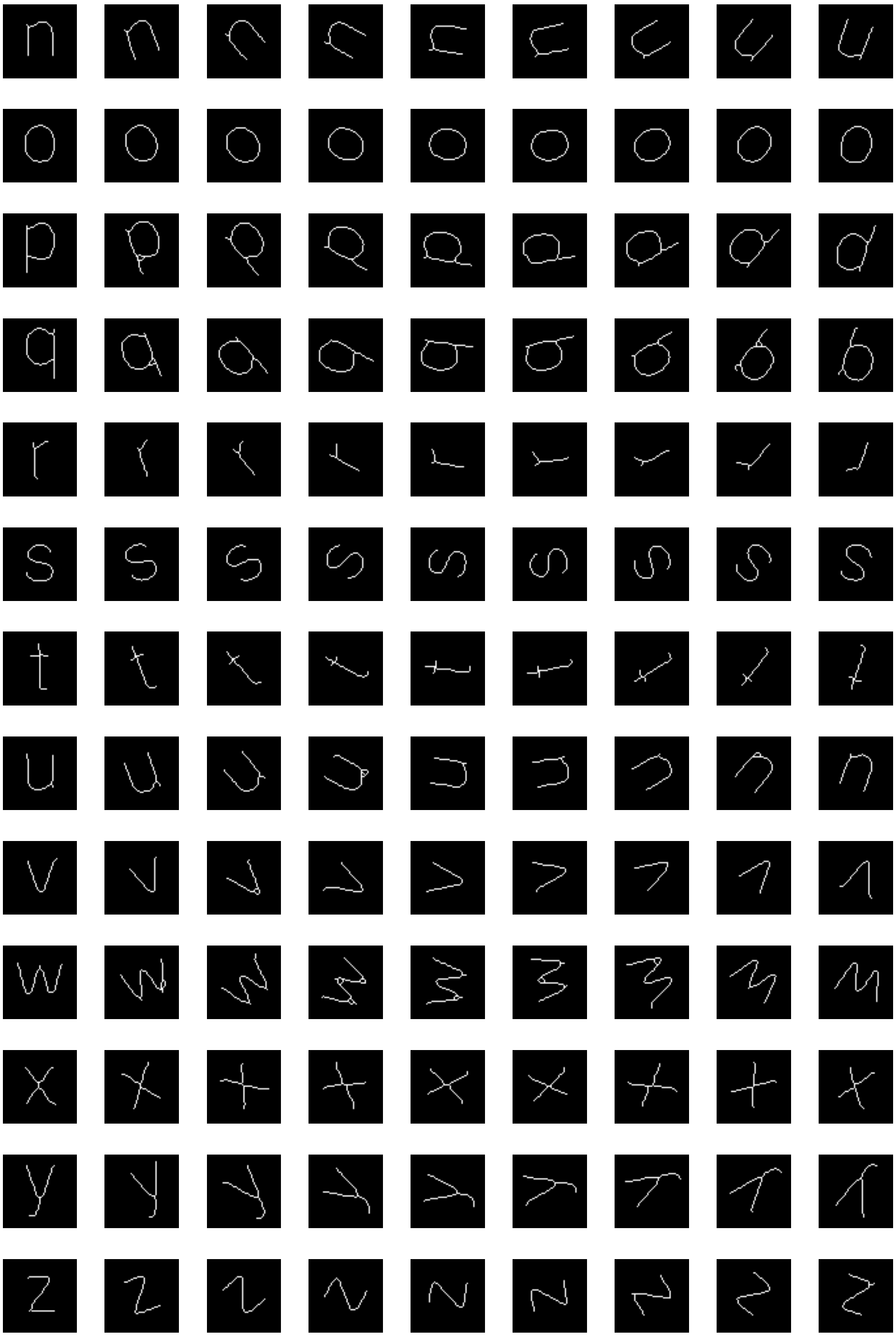


Figure 7.8: Training set of thinned, scanned characters (Part 2).

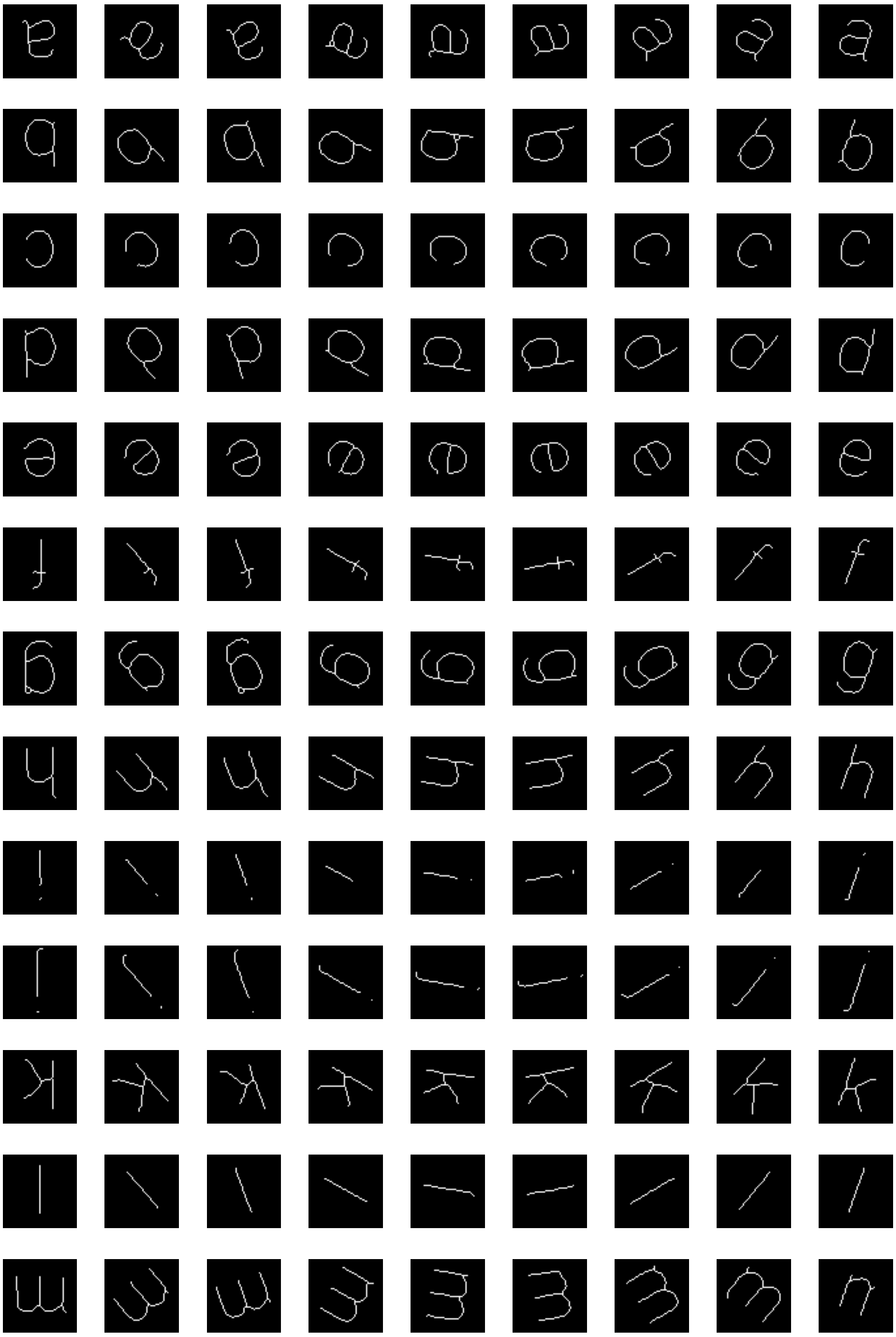


Figure 7.9: Test set of thinned, scanned characters (Part 1).

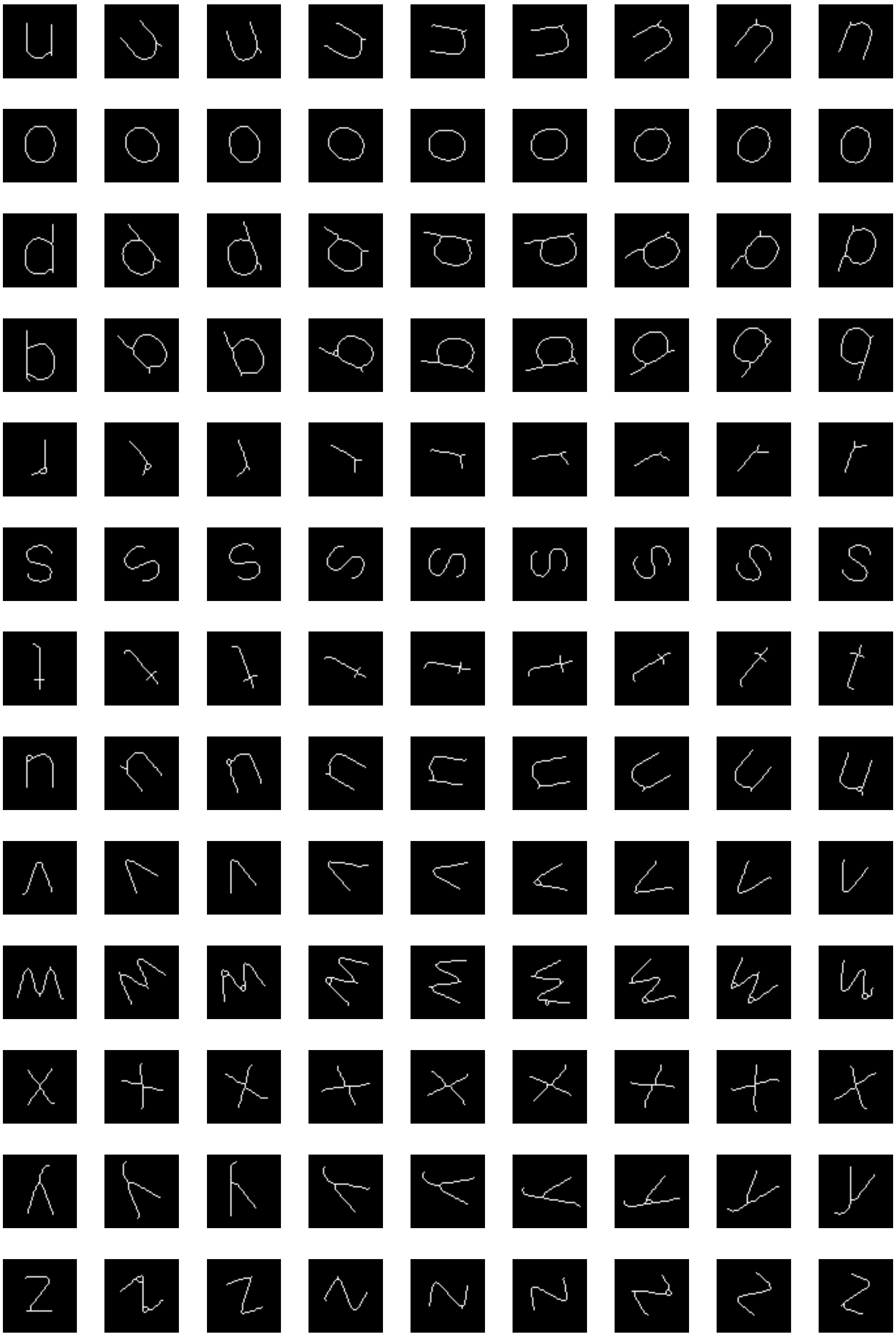
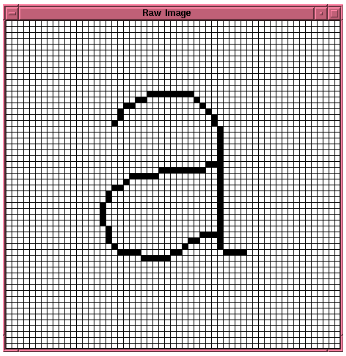
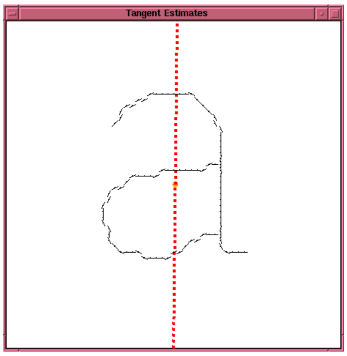


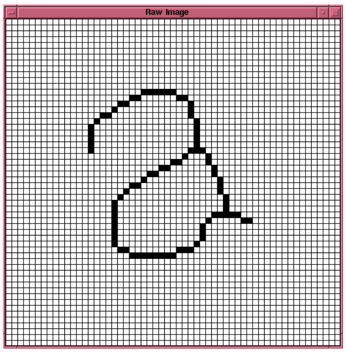
Figure 7.10: Test set of thinned, scanned characters (Part 2).



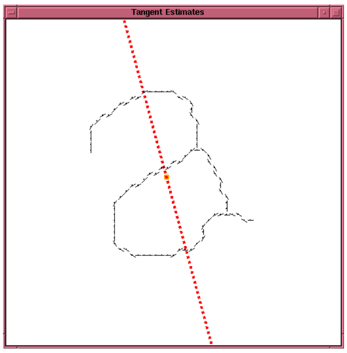
1. a00 Raw Image



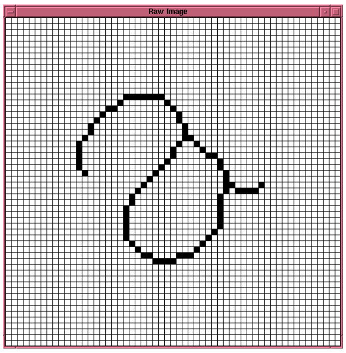
2. a00 Tangents



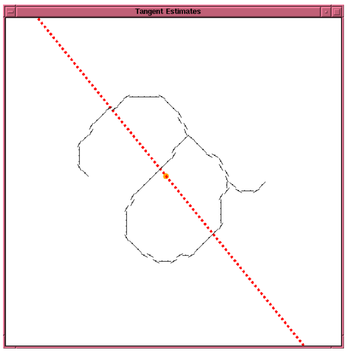
3. a01 Raw Image



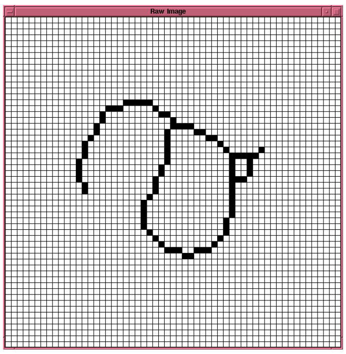
4. a01 Tangents



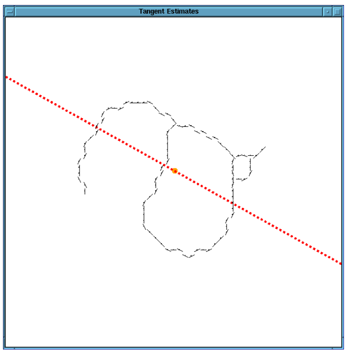
5. a02 Raw Image



6. a02 Tangents

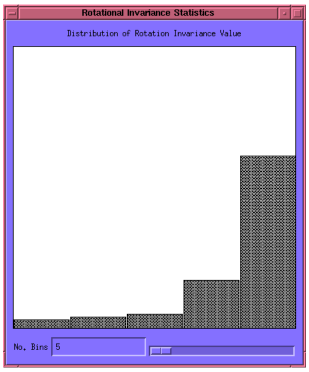


7. a03 Raw Image

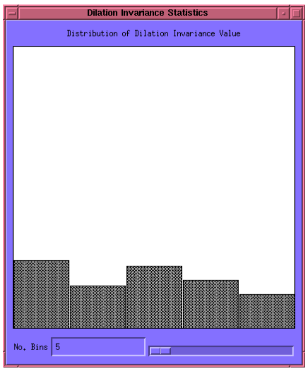


8. a03 Tangents

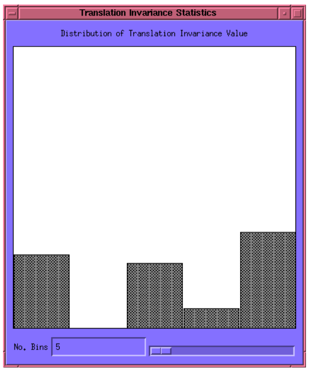
Figure 7.11: Tangents estimated for training examples of the letter "a".



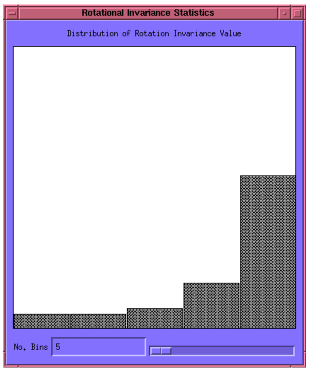

1. a00 Rotation



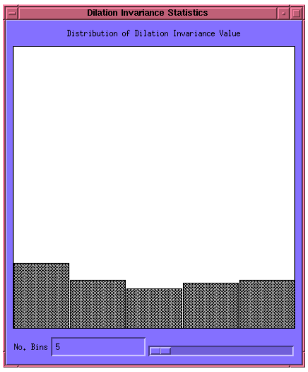

2. a00 Dilation



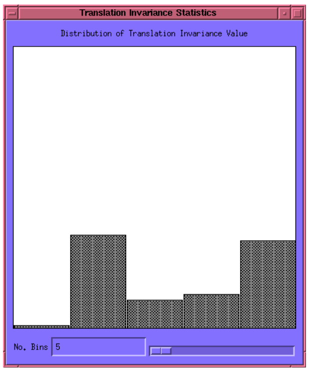

3. a00 Translation



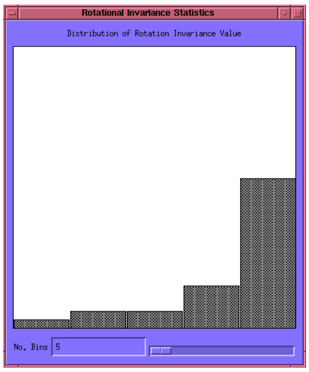

4. a01 Rotation



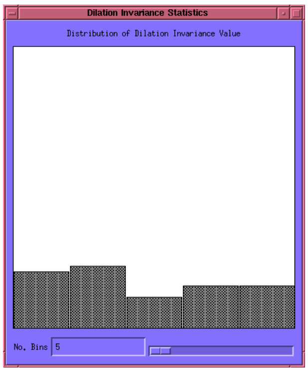

5. a01 Dilation



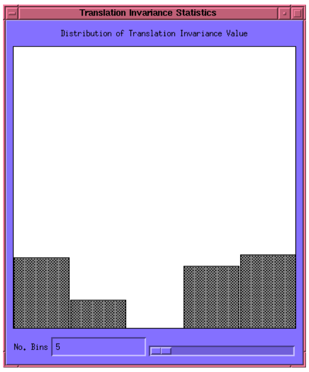

6. a01 Translation



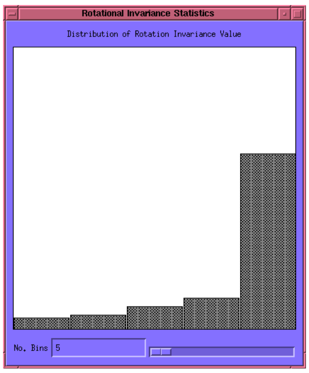

7. a02 Rotation



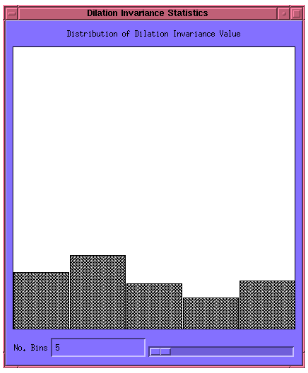

8. a02 Dilation



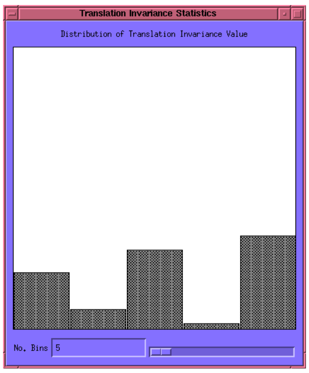

9. a02 Translation




10. a03 Rotation

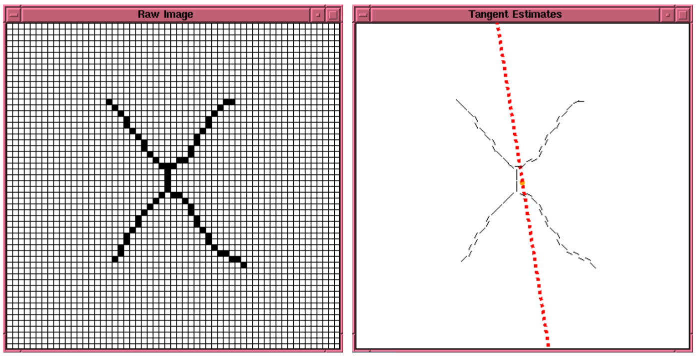



11. a03 Dilation

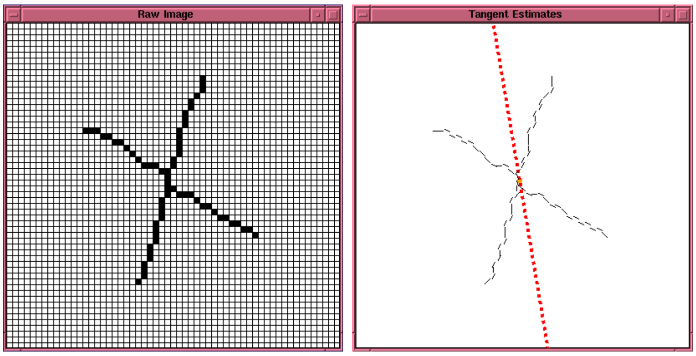



12. a03 Translation

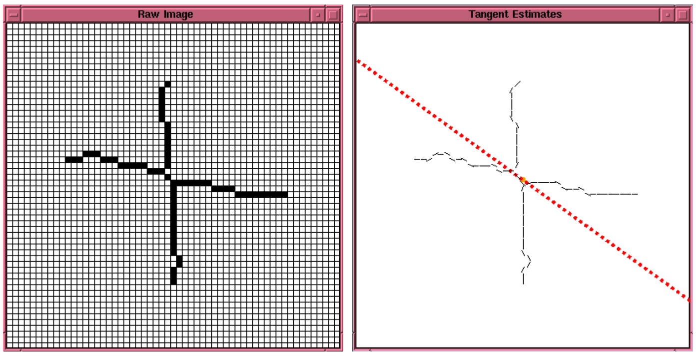
Figure 7.12: 5 bin Invariance Signatures for training examples of the letter “a”.



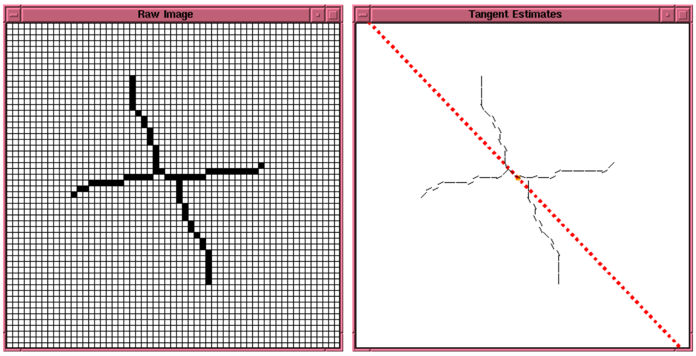

1. x00 Raw Image 2. x00 Tangents




3. x01 Raw Image 4. x01 Tangents

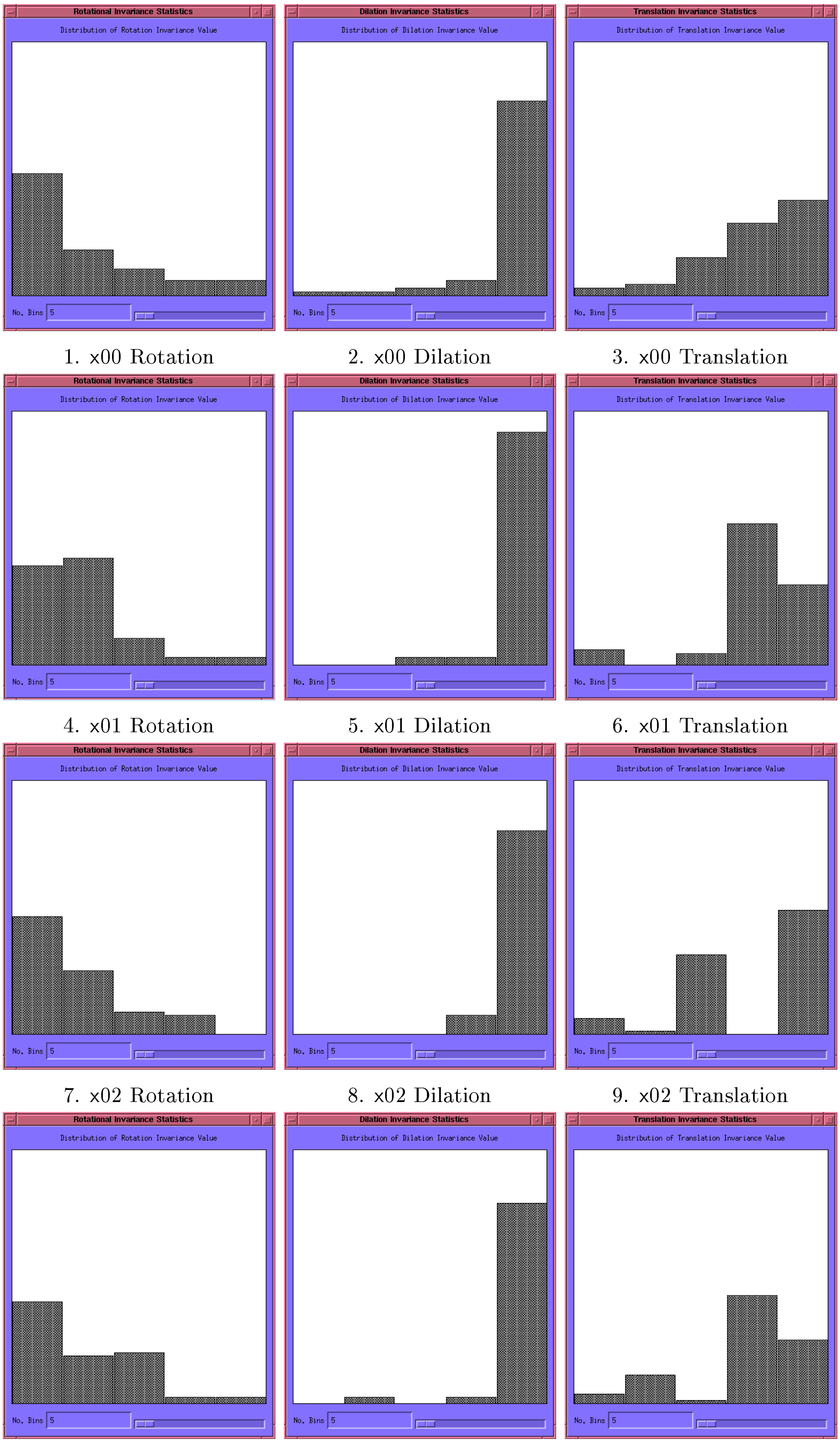



5. x02 Raw Image 6. x02 Tangents




7. x03 Raw Image 8. x03 Tangents

Figure 7.13: Tangents estimated for training examples of the letter “x”.



1. x00 Rotation 2. x00 Dilation 3. x00 Translation

4. x01 Rotation 5. x01 Dilation 6. x01 Translation

7. x02 Rotation 8. x02 Dilation 9. x02 Translation

10. x03 Rotation 11. x03 Dilation 12. x03 Translation

Figure 7.14: 5 bin Invariance Signatures for training examples of the letter “x”.